ESMA

Ecole Supérieure de Marrakech

3ème Années Génie Informatique

PHP

Pr. Abdelali El Gourari

PHP – Un site dynamique c'est quoi?

En général, on peut regrouper les sites Web en deux catégories: **les sites statiques** et **les sites dynamiques**.

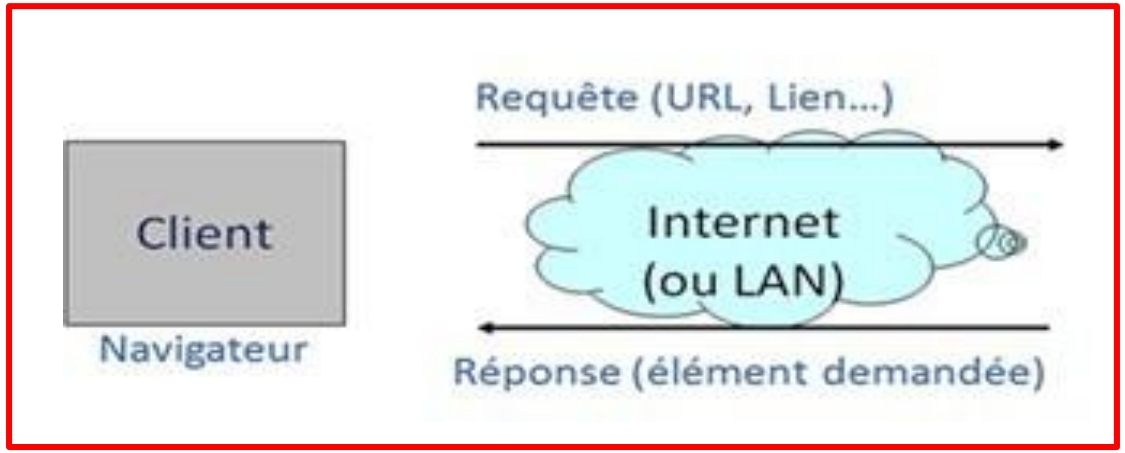
- Un site statique renferme un contenu figé qui ne change pas automatiquement et qui reste le même tant que le Webmaster n'est pas intervenu pour le modifier manuellement.

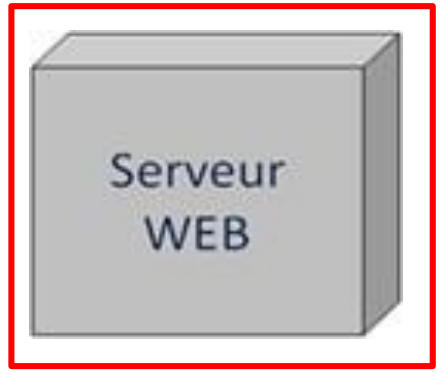
- Les sites dynamiques, quant-à eux, sont des sites Web dont le contenu change d'une manière autonome. Celui ci peut changer en fonction de la date, le navigateur utilisé par le client, la position géographique de celui-ci, les privilèges attribués à chaque utilisateur suite à une authentification par exemple, l'historique de navigation etc...

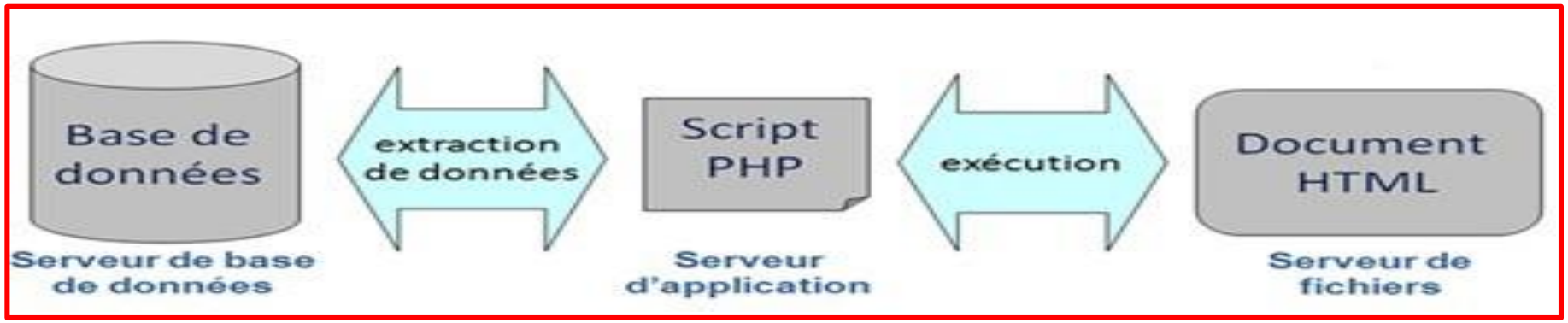
Les sites dynamiques reposent sur des langages dits **CGI** (pour Common Gateway Interface) dont PHP fait parti.

Généralités

PHP – De quoi aura-t-on besoin pour coder en PHP?







Introduction

De quoi aura-t-on besoin pour coder en PHP?

Préparation du poste

- ***WampServer, XAMP, MAMP…***
- ***VS code, VS, Notepad++,…***

1. Installation XAMP.

2. Démarrage d' **Apache et MySQL**.

3. Création d'espace du travail, il faut aller à C:\xampp**htdocs** et créez un dossier comme vous le souhaitez.

4. Ensuite, ouvrez ce dossier et créez un fichier dont le nom est **index.php**

Introduction

A quoi ressemble un document PHP?

- Une page PHP est suffixée par l'extension **.php**, mais cela ne veut pas dire qu'elle contient uniquement du code PHP. En effet, elle peut renfermer toutes les syntaxes que nous avons vu jusqu'ici à savoir: **HTML, CSS** et **JavaScript**.

- Il est donc tout à fait possible que votre page PHP contienne 4 langages différents à la fois.

```
<!DOCTYPE html>
<html>
 <body>
  <h1>My first PHP page</h1>
  <?php echo "Hello World!"; ?>
 </body>
</html>
```

Introduction

Dans l'exemple ci-dessous, les trois instructions d'écho ci-dessous sont égales et légales :

ECHO est le même que echo:

```
<!DOCTYPE html>
<html>
    <body>
        <?php
            ECHO “My name is Abdelali!<br>";
            echo "My name is Abdelali!<br>";
            EcHo "My name is Abdelali!<br>";
        ?>
    </body>
</html>
```

Regardez l'exemple ci-dessous ; seule la première instruction affichera la valeur de la $color variable ! En effet $color, $COLOR, et $color sont traitées comme trois variables différentes :

```
<!DOCTYPE html>
<html>
    <body>
    <?php

        $color = "red";
        echo "My car is " . $color . "<br>";
        echo "My house is " . $COLOR . "<br>";
        echo "My boat is " . $coLOR . "<br>";
    ?>
    </body>
</html>
```

Integration du code PHP

Les commentaires

- En PHP on peut intégrer des commentaires qui seront ignorés lors de l'exécution du script par le serveur. Comme pour JavaScript (ou pour le langage C) les commentaires en PHP peuvent avoir deux formes: Commentaire de fin de ligne: il s'agit d'un commentaire qui s'étend jusqu'à la fin de la ligne à partir du symbole double slash (//).
- Commentaire sur plusieurs lignes: il s'agit d'un bloc qui peut contenir plusieurs lignes comprises entre les symboles /* et */.

```
<?php
    // Commentaire de fin de ligne
    /*
        Bloc entier
        vu comme un
        commentaire
    */
?>
```

Integration du code PHP

Les variables

Une variable peut avoir un nom court (comme $x et $y) ou un nom plus descriptif ($age, $carname, $total_volume).

Règles pour les variables PHP :

- ✓ Une variable commence par le $ signe, suivi du nom de la variable
- ✓ Un nom de variable doit commencer par une lettre ou le caractère de soulignement
- ✓ Un nom de variable ne peut pas commencer par un nombre
- ✓ Un nom de variable ne peut contenir que des caractères alphanumériques et des traits de soulignement (Az, 0-9 et _).
- ✓ Les noms de variables sont sensibles à la casse ($age et $AGE sont deux variables différentes)

```php
<?php
    $a=10;    // Juste
    $_a9=true;    // Juste
    $9a="Bonjour";    // Faux (le chiffre ne doit pas figurer au début)
    $a b=5.3;       // Faux (le nom de la variable ne doit pas contenir d'espaces)
?>
```

Integration du code PHP

Les variables scalaires

```
$txt = “Abdelali";
echo " My name is $txt!";
```

```
$txt = “Abdelali";
echo " My name is" .$txt. "!";
```

- L'exemple suivant affichera la somme de deux variables :

```
$x = 5;
$y = 4;
echo $x + $y;
```

- PHP n'a pas de **commande** pour déclarer une variable, et le type de données dépend de la valeur de la variable.

```
$x = 5; // $x is an integer
$y = “Abdelali"; // $y is a string
echo $x;
echo $y
```

- La fonction renvoie var_dump() le type de données et la valeur :

```
$x = 5;
var_dump($x);
```

- Vous pouvez attribuer la même valeur à plusieurs variables sur une seule ligne :

```
$x = $y = $z = “Abdelali";
```

Integration du code PHP

L'instruction echo PHP

- L'instruction echo peut être utilisée avec ou sans parenthèses : echo ou echo().

```
echo "Hello"; //same as: echo("Hello");
```

L'exemple suivant montre comment générer du texte avec la commande echo (notez que le texte peut contenir du balisage HTML) :

```
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters."
```

L'exemple suivant montre comment générer du texte et des variables avec l'instruction echo :

```
$txt1 = “The best names in the world";
$txt2 = "Abdelali";
echo "<h2>$txt1</h2>";
echo "<p> My name is $txt2</p>";
```

```
$txt1 = “The best names in the world";
$txt2 = "Abdelali";
echo "<h2>" .$txt1. "</h2>";
echo "<p> My name is" .$txt2. "</p>";
```

Integration du code PHP

L'instruction d'impression PHP

- L'instruction print peut être utilisée avec ou sans parenthèses : print ou print().

```
print "Hello"; //same as: print("Hello");
```

L'exemple suivant montre comment générer du texte avec la commande print (notez que le texte peut contenir du balisage HTML) :

```
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!<br>";
print "This ", "string ", "was ", "made ", "with multiple parameters."
```

L'exemple suivant montre comment générer du texte et des variables avec l'instruction print :

```
$txt1 = “The best names in the world";
$txt2 = "Abdelali";
print "<h2>$txt1</h2>";
print "<p> My name is $txt2</p>";
```

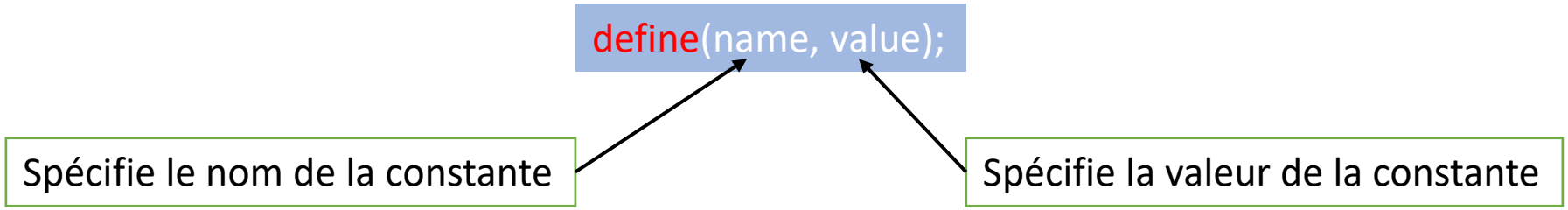
```
$txt1 = “The best names in the world";
$txt2 = "Abdelali";
print "<h2>" .$txt1. "</h2>";
print "<p> My name is" .$txt2. "</p>";
```

Integration du code PHP

les constantes PHP

Les constantes servent aussi à stocker des valeurs dans un programme, mais à l'inverse des variables, leurs valeurs ne changent pas.

Pour créer une constante, utilisez la fonction **define()**.



Exemple

```
define("Name", "My Name is Abdelali");
echo Name;
```

```
Const Constant  = 'Abdelali';
```

Exemple

```
echo Name;
```

Integration du code PHP

les constantes PHP

- ***Tableaux de constantes PHP***

À partir de PHP7, vous pouvez créer une constante de tableau à l'aide de la fonction define().

```
define(“Names", ["Ali", “Ahmed", “Sara"]);
echo Names[0];
```

Les constantes sont automatiquement globales et peuvent être utilisées dans l'ensemble du script.

Exemple:

```
define(“Name", " My Name is Abdelali");

function abc() {
  echo Name;
}

abc();
```

Integration du code PHP

les opérateurs

Les **opérateurs** sont des **symboles** qui permettent de faire des opérations sur les variables. Les opérateurs sont souvent les mêmes dans la plupart des langages de programmation et ils sont représentés par des symboles similaires dans la plupart des cas.

En **PHP** on distingue 5 familles d'opérateurs:

- ✓ **Opérateurs arithmétiques.**
- ✓ **Opérateurs d'incrémentation.**
- ✓ **Opérateurs assignement (affectation).**
- ✓ **Opérateurs de comparaison.**
- ✓ **Opérateurs logiques.**

Integration du code PHP

les opérateurs

- ***Opérateurs arithmétiques en PHP***

Les opérateurs arithmétiques en PHP sont utilisés avec des valeurs numériques pour effectuer des opérations arithmétiques courantes, telles que l'addition, la soustraction, la multiplication, etc.

Operator	Name	Example
+	Addition	$x + $y
-	Subtraction	$x - $y
*	Multiplication	$x * $y
/	Division	$x / $y
%	Modulus	$x % $y
**	Exponentiation	$x ** $y

```php
<?php
$x = 10;
$y = 2;

echo $x ** $y;
?>
```

100

- ***Opérateurs d'affectation PHP***

L'opérateur d'affectation de base en PHP est « = ». Cela signifie que l'opérande de gauche est défini sur la valeur de l'expression d'affectation de droite.

Assignment	Same as...
x = y	x = y
x += y	x = x + y
x -= y	x = x - y
x *= y	x = x * y
x /= y	x = x / y
x %= y	x = x % y

Integration du code PHP

les opérateurs

- ***Opérateurs de comparaison***

Les opérateurs de comparaison PHP sont utilisés pour comparer deux valeurs (nombre ou chaîne) :

Operator	Name	Example
==	Equal	$x == $y
===	Identical	$x === $y
!=	Not equal	$x != $y
<>	Not equal	$x <> $y
!==	Not identical	$x !== $y
>	Greater than	$x > $y
<	Less than	$x < $y
>=	Greater than or equal to	$x >= $y
<=	Less than or equal to	$x <= $y
<=>	Spaceship	$x <=> $y

- ***Opérateurs d'incrémentation/décrémentation***

Les opérateurs de décrémentation PHP sont utilisés pour décrémenter la valeur d'une variable.

Operator	Same as...
++$x	Pre-increment
$x++	Post-increment
--$x	Pre-decrement
$x--	Post-decrement

Integration du code PHP

les opérateurs

- ***Opérateurs logiques***

Les opérateurs logiques PHP sont utilisés pour combiner des instructions conditionnelles.

Operator	Name
and	And
or	Or
xor	Xor
&&	And
||	Or
!	Not

- ***Opérateurs d'incrémentation/décrémentation***

Les opérateurs de tableau PHP sont utilisés pour comparer des tableaux.

Operator	Name
+	Union
==	Equality
===	Identity
!=	Inequality
<>	Inequality
!==	Non-identity

Les structures de contrôle

Les structures conditionnelles

Très souvent, lorsque vous écrivez du code, vous souhaitez effectuer différentes actions pour différentes conditions. Vous pouvez utiliser des instructions conditionnelles dans votre code pour ce faire.

En PHP, nous avons les instructions conditionnelles suivantes :

Instruction if - exécute du code si une condition est vraie
Instruction if...else - exécute un code si une condition est vraie et un autre code si cette condition est fausse
Instruction if...elseif...else - exécute des codes différents pour plus de deux conditions
Instruction switch - sélectionne l'un des nombreux blocs de code à exécuter

Les structures de contrôle

Les Instructions Conditionnelles

- ***L'instruction if***

```
if (condition) {
  // I am very happy !!!!;
}
```



```
if (20 > 7) {
  echo “Ohm! Nice";
}
```

Nous pouvons également utiliser des variables dans l'instruction if :

```
$x = 10;
if ($x < 100) {
echo " Ohm! Nice";
}
```

Les structures de contrôle

Les structures conditionnelles

- ***Opérateurs de comparaison***

Vérifiez si $x est égal à 28 :

```
$x = 28;
if ($x == 28) {
echo "Ohm! Nice!";
}
```

Vérifiez si $x est supérieur à $x, et si $x est inférieur à $y:

```
$x = 100;
$y = 14;
$z = 200;
if ($x > $y && $x < $z ) {
echo “It is true";
}
```

Vérifiez si $x c'est 1, 2, 3, 4, 5 ou 6 :

```
$x = 5;
if ($x == 0 || $x == 1 || $x == 3 || $x == 4 || $x == 5 || $x == 6) {
echo "$x is a number between 0 and 6";
}
```

Les structures de contrôle

Les structures conditionnelles

- ***L'instruction if...else***

L'instruction **if...else** exécute du code si une condition est vraie, et un autre code si cette condition est fausse.

```
if (condition) {
  // I am very happy !!!!;
} else {
  // I am so sad !!!;
}
```

```
$x = “Ali";
if ($x == “Ali") {
echo “Yes, I am Ali!"; }
else {
echo “No, I am not Ali!";
}
```

- ***L'instruction if...elseif...else***

```
if (condition) {
// I am very happy !!!!;}
elseif (condition) {
// I am so sad !!!;}
else {
Oops !!!!!;
}
```

```
$x = “Ali";
if ($x == “Ali") {
echo “Yes, I am Ali!"; }
elseif ($x == “Ahmed") {
echo “No, I am Ahmed!"; }
else {
echo “Your name is not in our database!"; }
```

Les structures de contrôle

Les structures conditionnelles

- ***Déclaration en une seule ligne :***

```
$x = 10;

if ($x < 10) $y = “Ali";

echo $y
```

- ***Déclaration sur une seule if et else :***

```
$x = 10;

$x = $y < 10 ? “Ali" : “Not Ali";

echo $x;
```

Les structures de contrôle

Les structures conditionnelles

Utilisez l' instruction switch pour sélectionner l'un des nombreux blocs de code à exécuter .

```
switch (expression) {
    case label1: //code;
    break;
    case label2: //code ;
    break;
    case label3: //code;
    break;
    default: //code;
}
```

```
$favcolor = "red";
switch ($favcolor) {
    case "red": echo "Your favorite color is red!";
    break;
    case "blue": echo "Your favorite color is blue!";
    break;
    case "green": echo "Your favorite color is green!";
    break;
    default: echo "Your favorite color is neither red, blue, nor green!"; }
```

```
$d = 3;
switch ($d) {
case 1: case 2: case 3: case 4: case 5: echo "The weeks feels
so long!";
break;
case 6: case 0: echo "Weekends are the best!";
break;
default: echo "Something went wrong"; }
```

Les structures répétitives

Les boucles

Boucle for

```
for (expression1, expression2, expression3) {
// code
}
```

Imprimez les nombres de 0 à 10 :

```
for ($x = 0; $x <= 10; $x++) {
echo "The number is: $x <br>";
}
```

Arrêtez la boucle quand $x = 3 :

```
for ($x = 0; $x <= 10; $x++) {
if ($x == 3) break;
echo "The number is: $x <br>";
}
```

Structure while

```
$i = 1;
while ($i < 6) {
     echo $i;
$i++;
}
```

```
$i = 1;
while ($i < 6) {
     if ($i == 3)
     break;
     echo $i;
     $i++;
}
```

```
$i = 0;
while ($i < 6) {
     $i++;
     if ($i == 3)
     continue;
     echo $i;
}
```

Les structures répétitives

Les boucles

Boucle for

Arrêtez la boucle et passez à l'itération suivante si $x = 3 :

```
for ($x = 0; $x <= 10; $x++) {
  if ($x == 3) continue;
  echo "The number is: $x <br>";
}
```

Structure do while

Imprimez $i tant que $i est moins de 6 :

```
$i = 1;

do {
  echo $i;
  $i++;
} while ($i < 6);
```

Structure do while

```
$i = 1;

do {
  if ($i == 3) break;
  echo $i;
  $i++;
} while ($i < 6);
```

```
$i = 0;

do {
  $i++;
  if ($i == 3) continue;
  echo $i;
} while ($i < 6);
```

Exercices

Exercice 1:
Écrivez un programme pour effectuer la somme ou l'addition de deux nombres en programmation PHP. Vous pouvez utiliser des variables et des opérateurs en PHP

Exercice 2:
Ecrire un programme PHP pour calculer la facture d'électricité en utilisant des conditions ***if-else***.

Conditions:

- Pour les 50 premières unités – Rs. 3,50/unité ➔ (***unités<=50)***
- Pour les 100 prochaines unités – Rs. 4,00/unité ➔ (***unités>50 && unités<=100)***
- Pour les 100 prochaines unités – Rs. 5,20/unité ➔(***unités>100 && unités<=200)***
- Pour les unités supérieures à 250 – Rs. 6,50/unité

Exercice 3:
En utilisant la boucle **for**, afficher la table de multiplication du chiffre 7.

Exercices

Exercice 3:

Ecrire un programme qui lit au clavier les valeurs de trois résistances et de trois capacités et calcule leur résistance et leur capacité équivalente, respectivement, dans les deux cas :

- Les trois résistances et les trois capacités sont placées en série
- Les trois résistances et les trois capacités sont placées en parallèle

Les résultats doivent être affichés dans chaque cas.

Exercice 4:

Ecrire un programme permettant de calculer la valeur de l'expression E,

$$E = 1 + \frac{1}{1+2} + \frac{1}{1+2+3} + \cdots + \frac{1}{1+2+3+\cdots+N} \quad , \quad \text{avec } (N \geq 2).$$

Gestion des formulaires en PHP

Les superglobales $_GET et $_POST en PHP sont utilisées pour collecter des données de formulaire.

```
<html>
<body>

    <form action="home.php" method="POST">
        Name: <input type="text" name="name"><br>
        E-mail: <input type="text" email ="email"><br>
        <input type="submit">
    </form>

</body>
</html>
```

Lorsque l'utilisateur remplit le formulaire ci-dessus et clique sur le bouton d'envoi, les données du formulaire sont envoyées pour traitement dans un fichier PHP nommé «home.php ». Les données du formulaire sont envoyées avec la méthode HTTP POST.

Pour afficher les données soumises, vous pouvez simplement répéter toutes les variables.

Gestion des formulaires en PHP

home.php

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>

</body>
</html>
```

```
<html>
<body> Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>
</body>
</html>
```

Les tableaux

En PHP, **un tableau** est une structure de données qui peut contenir une **collection de valeurs**, telles que **des nombres**, **des chaînes de caractères**, **des objets**, ou même d'autres tableaux.

Tableaux indexés:
Les tableaux **indexés** sont des tableaux où **chaque élément** est **associé à un indice numérique**, commençant généralement par **zéro**.

```
// Création d'un tableau indexé

$fruits = array('Pomme', 'Banane', 'Orange');

// Accès aux éléments du tableau
        echo $fruits[0]; // Affiche "Pomme"
        echo $fruits[1]; // Affiche "Banane"
        echo $fruits[2]; // Affiche "Orange"
```

Les tableaux

Tableaux associatifs:

Les tableaux associatifs sont des tableaux où chaque élément est associé à une **clé**.

$Key => $value

```php
// Création d'un tableau associatif

$personne = array(
        'nom' => ‘El Gourari',
        'prénom' => ‘Abdelali',
        'âge' => 27
);
// Accès aux éléments du tableau
echo $personne['nom'];          // Affiche “El Gourari"
echo $personne['prénom'];       // Affiche “Abdelali"
echo $personne['âge'];          // Affiche 27
```

Les tableaux

Ajout et modification d'éléments dans un tableau

```php
// Création d'un tableau

$couleurs = array('rouge', 'bleu', 'vert');

// Ajout d'un élément à la fin du tableau
$couleurs[] = 'jaune';

// Modification d'un élément
$couleurs[1] = 'orange';
```

Parcourir un tableau avec foreach

```php
$notes = array(10, 15, 12, 18);

// Afficher chaque élément du tableau
foreach ($notes as $note) {
        echo $note . '<br>';
}
```

Ces exemples illustrent quelques notions de base sur les tableaux en PHP. Les tableaux sont des structures de données extrêmement polyvalentes et sont utilisés très fréquemment dans le développement web avec PHP.

Les tableaux

Fonctions utiles pour les tableaux

PHP dispose de nombreuses fonctions intégrées pour travailler avec les tableaux, telles que **count, array_push, array_pop, array_shift, array_unshift, sort, rsort, array_keys, array_values,** etc.

```
fruits = array('Pomme', 'Banane', 'Orange');

// Nombre d'éléments dans le tableau
        echo count($fruits); // Affiche 3
// Ajout d'un élément à la fin du tableau
        array_push($fruits, 'Fraise');
// Suppression du dernier élément du tableau
        array_pop($fruits);
// Affichage des clés du tableau
        print_r(array_keys($fruits));
// Affichage des valeurs du tableau
        print_r(array_values($fruits));
// Tri du tableau sort($fruits);
        print_r($fruits);
```

Les fonctions

En PHP, une fonction est un bloc de code nommé qui peut être réutilisé pour effectuer une tâche spécifique. Les fonctions peuvent accepter des arguments en entrée et renvoyer une valeur en sortie, mais elles peuvent aussi ne rien renvoyer du tout.

Déclaration d'une fonction

```php
function direBonjour() {

echo "Bonjour !";
}
```

Appel d'une fonction

```php
direBonjour(); // Affiche "Bonjour !"
```

Fonctions avec des arguments

```php
function direBonjourPersonne($nom) {
echo "Bonjour, $nom !";
}
direBonjourPersonne("Ali");

// Affiche "Bonjour, Ali !"
```

Fonctions avec une valeur de retour

```php
function addition($a, $b) {
return $a + $b;
}

$resultat = addition(5, 3);

// $resultat vaut 8
```

Les fonctions

Fonctions avec des arguments par défaut:

```
function direBonjour($nom = "mon ami") {
echo "Bonjour, $nom !";
}
direBonjour(); // Affiche "Bonjour, mon ami !"
direBonjour("Ali"); // Affiche "Bonjour, Ali !"
```

Fonctions récursives:

Les fonctions récursives sont des fonctions qui s'appellent elles-mêmes.

```
function factorielle($n) {
if ($n == 0 || $n == 1) {
            return 1; }
else {
            return $n * factorielle($n - 1);
}
}
echo factorielle(5); // Affiche 120 (car 5! = 5*4*3*2*1)
```

Les fonctions

Fonctions anonymes (fonctions lambda)

Les fonctions anonymes sont des fonctions sans nom qui peuvent être assignées à une variable ou utilisées comme argument de fonction.

```
$saluer = function($nom) {
echo "Bonjour, $nom !";
};
$saluer("Ali"); // Affiche "Bonjour, Ali !"
```

Ces exemples illustrent quelques notions de base sur les fonctions en PHP. Les fonctions sont des éléments fondamentaux du langage qui permettent d'organiser et de réutiliser le code de manière efficace.

Les fonctions

Exercice : Calculer la moyenne des notes

Dans cet exercice, vous allez créer un script PHP qui prend un tableau de notes et calcule leur moyenne.

1. Créez un tableau contenant des notes (par exemple, [10, 15, 12, 18]).
2. Écrivez une fonction en PHP pour calculer la moyenne des notes.
3. Utilisez cette fonction pour afficher la moyenne des notes.

Les chaînes de caractères

En PHP, les chaînes de caractères sont utilisées pour représenter du texte. Voici quelques opérations de base que vous pouvez effectuer sur les chaînes de caractères :

Déclaration de chaînes de caractères

```
$texte = "Bonjour, monde!";


$texte2 = 'Ceci est une chaîne de caractères.';
```

Concaténation de chaînes de caractères

```
$nom = « Ali";
$salutation = "Bonjour, " . $nom . "!";
// Ou
$salutation = "Bonjour, $nom!";
```

Longueur d'une chaîne de caractères

```
$texte = "Hello";

$longueur = strlen($texte); // $longueur vaut 5
```

Recherche dans une chaîne de caractères

```
$texte = "Bonjour, monde!";
$position = strpos($texte, "monde");
// $position vaut 9
```

Les chaînes de caractères

Remplacement dans une chaîne de caractères

```php
$texte = "Bonjour, monde!";
$texte_modifie = str_replace("monde", "PHP", $texte);
// $texte_modifie vaut "Bonjour, PHP!"
```

Extraction d'une sous-chaîne

```php
$texte = "Bonjour, monde!";
$sous_chaine = substr($texte, 8, 5);
// $sous_chaine vaut "monde"
```

Conversion de cas

```php
$texte = "Bonjour";
$texte_majuscule = strtoupper($texte);
// $texte_majuscule vaut "BONJOUR"
$texte_minuscule = strtolower($texte);
// $texte_minuscule vaut "bonjour"
```

Ces exemples couvrent certaines opérations courantes que vous pouvez effectuer sur les chaînes de caractères en PHP. Les chaînes de caractères sont utilisées très fréquemment dans le développement web pour manipuler et afficher du texte sur les pages web.

Les chaînes de caractères

Comparaison de chaînes de caractères

```
$chaine1 = "Bonjour";
$chaine2 = "bonjour";  → ‘BONJOUR’ >’bonjour’ Dans le code ASCII
$resultat = strcmp($chaine1, $chaine2);
// $resultat vaut ‘‘-1’’ OU bien ‘’true’’ (différentes, mais pas sensibles à la casse)
```

Division d'une chaîne en un tableau

```
$texte = "Ceci est une phrase.";
$mots = explode(" ", $texte);
// $mots est un tableau ["Ceci", "est", "une", "phrase."]
```

Jointure de tableau en une chaîne

```
$mots = array("Ceci", "est", "une", "phrase.");
$texte = implode(" ", $mots);
// $texte vaut "Ceci est une phrase."
```

les sessions

En PHP, une **session** est une **façon de stocker des données côté serveur** pour un utilisateur donné sur **une période de temps définie**, généralement **entre plusieurs** pages ou **requêtes**. Cela permet de stocker des informations spécifiques à un utilisateur, telles que des **identifiants de connexion**, **des préférences utilisateur,** des paniers d'achat, etc.

Création de session en PHP

```php
<?php

// Démarre la session
session_start();
// Définit des variables de session
$_SESSION['utilisateur'] = ‘Abdelali';
$_SESSION['role'] = ‘Teacher';
echo 'Session créée et variables définies.';

?>
```

Utilisation de session en PHP

```php
<?php
// Démarre la session
session_start();
// Accède aux variables de session
$utilisateur = $_SESSION['utilisateur'];
$role = $_SESSION['role'];
echo 'Utilisateur : ' . $utilisateur . '<br>';
echo 'Rôle : ' . $role;

?>
```

les sessions

Dans cet exemple, **la première page** définit des variables de session pour l'utilisateur, puis l**a deuxième page** y accède pour afficher les informations stockées. Assurez-vous d'appeler **session_start()** au début de chaque script PHP où vous souhaitez accéder aux variables de session.

Exemple 1: Authentification d'utilisateur

```php
<?php
session_start();

// Vérification des identifiants
if ($_POST['username'] == 'Abdelali' && $_POST['password'] == '123') {
        $_SESSION['logged_in'] = true;
        $_SESSION['username'] = $_POST['username'];
        echo 'Vous êtes connecté en tant que ' . $_SESSION['username'];
}
else {
        echo 'Identifiants incorrects';
}
?>
```

les sessions

Exemple 2: Stockage de préférences utilisateur

```php
<?php session_start();

// Définition des préférences utilisateur
$_SESSION['preferences'] = array( 'theme' => 'dark', 'langue' => ‘Arab' );
echo 'Préférences utilisateur définies.';

?>
```

les sessions

Exemple 3: Gestion de panier d'achat

```php
<?php session_start();
// Ajout d'un produit au panier
if (isset($_GET['add_to_cart'])) {
        $product_id = $_GET['add_to_cart'];
        if (!isset($_SESSION['cart'][$product_id])) { $_SESSION['cart'][$product_id] = 1; }
        else { $_SESSION['cart'][$product_id]++; }
        echo 'Produit ajouté au panier.';
}
// Affichage du panier
echo '<h2>Panier d\'achat</h2>';
if (!empty($_SESSION['cart'])) {
        foreach ($_SESSION['cart'] as $product_id => $quantity) {
                echo 'Produit #' . $product_id . ' - Quantité: ' . $quantity . '<br>';
}
}
else { echo 'Le panier est vide.'; }
?>
```

les sessions

Ces exemples illustrent différentes façons d'utiliser les sessions en PHP, notamment pour l'authentification, le stockage de préférences utilisateur et la gestion d'un panier d'achat. Assurez-vous toujours de démarrer la session avec ***session_start()*** au début de chaque script où vous souhaitez utiliser les sessions.

Exemple 4: **Création d'un formulaire de connexion avec authentification:**

Dans cet exemple, nous allons créer un formulaire de connexion où l'utilisateur doit saisir un nom d'utilisateur et un mot de passe. Si les identifiants sont corrects, l'utilisateur sera redirigé vers une page de bienvenue, sinon un message d'erreur sera affiché. Il peut également se déconnecter en cliquant sur le lien de déconnexion, ce qui le ramènera à la page de connexion.

Les fonctions prédéfinies en PHP

En PHP, il y a de nombreuses fonctions prédéfinies qui facilitent le développement d'applications web et d'autres types de logiciels. Voici quelques-unes des fonctions prédéfinies les plus couramment utilisées :

Fonctions de manipulation de chaînes de caractères :

- **strlen()** : Retourne la longueur d'une chaîne de caractères.
- **strtolower()** : Convertit une chaîne de caractères en minuscules.
- **strtoupper()** : Convertit une chaîne de caractères en majuscules.
- **str_replace()** : Remplace une sous-chaîne par une autre dans une chaîne donnée.
- **substr()** : Extrait une sous-chaîne à partir d'une chaîne de caractères.

Les fonctions prédéfinies en PHP

Fonctions de manipulation de tableaux :

- ✓ **count()** : Retourne le nombre d'éléments dans un tableau.
- ✓ **array_push()** : Ajoute un ou plusieurs éléments à la fin d'un tableau.
- ✓ **array_pop()** : Supprime et retourne le dernier élément d'un tableau.
- ✓ **array_merge()** : Fusionne deux tableaux en un seul.
- ✓ **array_search()** : Recherche la valeur donnée dans un tableau et retourne la clé correspondante si elle est trouvée.

Fonctions de gestion des dates et heures :

- ✓ **date()** : Retourne la date et l'heure actuelles au format spécifié.
- ✓ **time()** : Retourne l'heure actuelle sous forme de **timestamp** UNIX. (1er janvier 1970)
- ✓ **strtotime()** : Convertit une chaîne de texte en timestamp UNIX. → 2024-05-09 →9 mai 2024

Les fonctions prédéfinies en PHP

Fonctions pour la gestion des fichiers :

- **file_get_contents()** : Lit le contenu d'un fichier dans une chaîne de caractères.
- **file_put_contents()** : Écrit une chaîne dans un fichier.
- **fopen()** : Ouvre un fichier.
- **fclose()** : Ferme un fichier ouvert.

Fonctions pour la gestion des sessions et des cookies :

- **session_start()** : Démarre une session PHP.
- **$_SESSION[]** : Tableau associatif pour stocker des variables de session.
- **setcookie()** : Définit un cookie à envoyer au client.

Les fonctions prédéfinies en PHP

Fonctions pour la manipulation des données :

- **json_encode()** : Convertit une variable PHP en format JSON.
- **json_decode()** : Convertit une chaîne JSON en objet ou tableau PHP.
- **serialize()** : Convertit une variable PHP en une représentation storable.
- **unserialize()** : Convertit une représentation sérialisée en variable PHP.

Ce ne sont là que quelques exemples parmi les nombreuses fonctions prédéfinies disponibles en PHP.

Les fonctions prédéfinies en PHP

En PHP, pour interagir avec une base de données, notamment les bases de données MySQL, il existe plusieurs fonctions prédéfinies qui permettent de réaliser des opérations telles que **la connexion à la base de données**, **l'exécution de requêtes SQL**, **la récupération des résultats**, etc. Voici quelques-unes des fonctions de gestion de base de données les plus couramment utilisées en PHP avec MySQL :

Connexion à la base de données :

- ✓ **mysqli_connect()** : Établit une connexion à la base de données MySQL.
- ✓ **mysqli_select_db()** : Sélectionne une base de données spécifique sur le serveur MySQL.
- ✓ **mysqli_close()** : Ferme la connexion à la base de données.

Les fonctions prédéfinies en PHP

Exemple de connexion à la base de données :

```php
$servername = "localhost";
$username = "utilisateur";
$password = "motdepasse";
$database = "basededonnees";


// Connexion
$conn = mysqli_connect($servername, $username, $password, $database);


// Vérifier la connexion
if (!$conn) {
        die("Connexion échouée : " . mysqli_connect_error());
}
echo "Connexion réussie";
```

Les fonctions prédéfinies en PHP

Exécution des requêtes SQL :

- ✓ **mysqli_query()** : Exécute une requête SQL sur la base de données.

Exemple d'exécution d'une requête SELECT :

```
$sql = "SELECT * FROM utilisateurs";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
        while($row = mysqli_fetch_assoc($result)) {
                echo "ID: " . $row["id"] . " - Nom: " . $row["nom"] . "<br>";
}
} else {
        echo "Aucun résultat trouvé";
}
```

Les fonctions prédéfinies en PHP

Récupération des résultats :

- ✓ **mysqli_fetch_assoc()** : Récupère une ligne de résultat sous forme de tableau associatif.
- ✓ **mysqli_fetch_array()** : Récupère une ligne de résultat sous forme de tableau indexé et associatif.
- ✓ **Mysql_num_rows()** : le nombre des lignes retournées par une requête **SELECT** exécutée sur une base de données **MySQL**.

Gestion des erreurs :

- ✓ **mysqli_error()** : Retourne une chaîne de description de l'erreur lorsqu'une requête échoue.

Ces fonctions de gestion de base de données en PHP vous permettent de réaliser des opérations de base comme la connexion, l'exécution de requêtes SQL, la récupération et l'affichage des résultats, et la gestion des erreurs.

Les fonctions prédéfinies en PHP

En PHP, il est crucial de mettre en place des mesures de **sécurité pour protéger votre application web** contre les attaques et les failles de sécurité. Voici quelques-unes des fonctions et bonnes pratiques de sécurité couramment utilisées en PHP :

Validation des données :

- ✓ **filter_var()** : Valide et filtre les données selon différents filtres (par exemple, **FILTER_VALIDATE_EMAIL** pour valider une adresse email).
- ✓ **htmlspecialchars()** : Convertit les caractères spéciaux en entités HTML pour éviter les attaques XSS (Cross-Site Scripting).
- ✓ **strip_tags()** : Supprime toutes les balises HTML et PHP d'une chaîne pour éviter les injections de code.

Les fonctions prédéfinies en PHP

Exemple de validation d'une adresse email :

```
$email = "contact@example.com";
if (filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Adresse email valide";
}
else {
        echo "Adresse email invalide";
}
```

Les fonctions prédéfinies en PHP

Prévention des injections SQL :

- ✓ Utilisation de requêtes préparées avec **mysqli_prepare()** et **mysqli_stmt_bind_param()** pour échapper les valeurs et éviter les injections SQL.

Exemple de requête préparée :

```
$stmt = mysqli_prepare($conn, "INSERT INTO utilisateurs (nom, email) VALUES (?, ?)");
mysqli_stmt_bind_param($stmt, "ss", $nom, $email);
mysqli_stmt_execute($stmt);
```

Les fonctions prédéfinies en PHP

Gestion des sessions :

- ✓ **session_start()** : Démarre une session PHP sécurisée pour stocker des variables de session.
- ✓ Utilisation de **session_regenerate_id()** pour régénérer l'identifiant de session afin de prévenir les attaques de fixation de session.

Exemple de démarrage de session :

```
session_start();
$_SESSION['utilisateur'] = ‘Ali';
```

Les fonctions prédéfinies en PHP

Cryptographie :

- ✓ **password_hash()** : Permet de hasher les mots de passe de manière sécurisée en utilisant l'algorithme **bcrypt**.
- ✓ **password_verify()** : Vérifie si un mot de passe correspond au hash stocké en base de données.

*Exemple d'utilisation de **password_hash()** :*

```
$mot_de_passe = "motdepasse123";
$hash = password_hash($mot_de_passe, PASSWORD_DEFAULT);
```

Les fonctions prédéfinies en PHP

Protection contre les attaques CSRF (Cross-Site Request Forgery) :

- ✓ Utilisation de **jetons CSRF** (tokens CSRF) dans les formulaires et vérification de ces jetons côté serveur pour valider les requêtes.

Exemple de génération et de vérification de jetons CSRF :

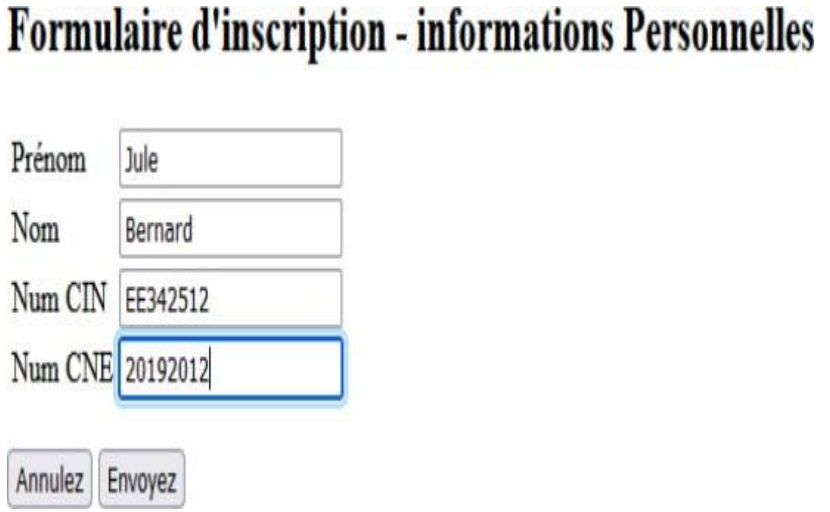
```php
// Génération du jeton CSRF
$jeton_csrf = bin2hex(random_bytes(32));
$_SESSION['jeton_csrf'] = $jeton_csrf;


// Vérification du jeton CSRF lors de la soumission du formulaire
if ($_POST['jeton_csrf'] === $_SESSION['jeton_csrf']) {
 // Traitement sécurisé
} else {
// Jeton CSRF invalide
}
```

Les fonctions prédéfinies en PHP

En utilisant ces fonctions et bonnes pratiques de sécurité en PHP, vous pouvez renforcer la sécurité de votre application web et réduire les risques d'attaques et de failles de sécurité.

Interface graphique Formulaire de saisie d'informations personnelles

```
<title>Formulaire de saisie</title>
</head>
<body>
<h2>Formulaire d'inscription - informations Personnelles</h2>
<form action="recup_infos.php" method="post">
<table>
<tr><td>Pr&eacute;nom</td><td><input type="text" name="prenom" /></td></tr>
<tr><td>Nom</td><td><input type="text" name="nom" /></td></tr>
<tr><td>Num CIN </td><td><input type="text" name="numCIN"/></td></tr>
<tr><td>Num CNE</td><td><input type="text" name="numCNE" /></td></tr>
</table>
<p>
<input type="reset" name="reset" value="Annulez" />
<input type="submit" name="formInfo" value="Envoyez" />
```



Récupération et affichage des données user

1

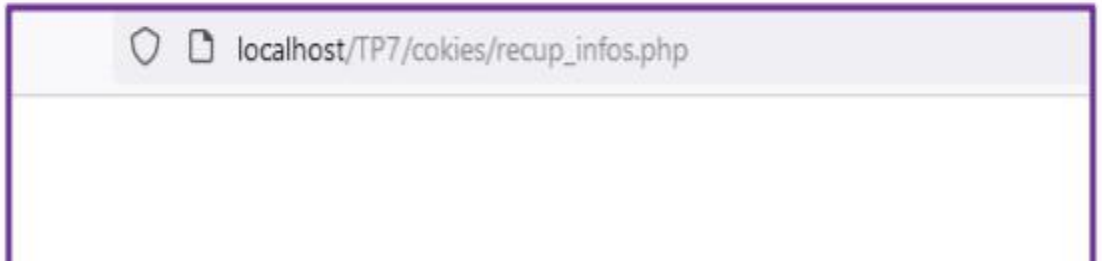
```php
<?PHP
if (isset($_POST['prenom']) && isset($_POST['nom']) && isset($_POST['numCIN']) &&
isset($_POST['numCNE'])){
echo "<h2>Informations personnelles:</h2>\n";
echo "<p>".$_POST['prenom']." ".$_POST['nom'].", on connait votre num&eacute;ro de carte: ";
echo $_POST['numCIN']." et votre numéro national d'&eacute;tudiant:
".$_POST['numCNE']."</p>\n";
```

Récupération et affichage des données entrées par l'utilisateur

Informations personnelles:

Jule Bernard, on connait votre numéro de carte: EE342512 et votre numéro national d'étudiant: 20192012

Exemple d'affichage des données entrées par l'utilisateur



Si on fait appel encore une fois à cette page sans passer par le formulaire, on va obtenir le résultat

Récupération et affichage des données user

2

```
setCookie('nom',$_POST['nom'],time()+10*60);
setCookie('prenom',$_POST['prenom'],time()+10*60);
setCookie('numCIN',$_POST['numCIN'],time()+10*60);
setCookie('numCNE',$_POST['numCNE'],time()+10*60);
}
```

Mémorisation des informations reçues dans un cookie avec une durée de vie de 10 minutes.

3

```
else{
if (isset($_COOKIE['prenom']) && isset($_COOKIE['nom']) &&
isset($_COOKIE['numCIN']) && isset($_COOKIE['numCNE'])){
echo "<h2>Information Personel:</h2>\n";
echo "<p>".$_COOKIE['prenom']." ".$_COOKIE['nom'].", on connait votre num&eacute;ro de carte: ";
echo $_COOKIE['numCIN']." et votre numéro national d'&eacute;tudiant: ".$_COOKIE['numCNE']."</p>\n";
echo "<p>Nous avons retrouv&eacute; ces informations personnel dans les cookies de votre ordinateur !!
}
```

Recherche des informations dans les données issues du formulaire dans les cookies de votre machine

4

Information Personel:

Jule Bernard, on connait votre numéro de carte: EE342512 et votre numéro national d'étudiant: 20192012

Nous avons retrouvé ces informations personnel dans les cookies de votre ordinateur !!!

Si encore une fois, on fait rappel à cette page sans passer par le formulaire, on va obtenir le résultat suivant (suite au cookie crée)

5

```
else{
echo "<h2>Informations Personnelles</h2>\n";
echo "<p>Elles nous sont inconnues car vous n'avez utilise ni le formulaire ni les cookies \n";
} }
?>
```

Objectif :
L'authentification avec les sessions

Afin de protéger les pages web et de s'assurer que l'utilisateur est authentifié, nous allons utiliser les sessions.
Pour répondre à ce besoin il faut noter que :

- Lors de l'authentification, les login et mots de passe seront mémorisés dans des variables de sessions
- A chaque page accédée, la vérification de la connaissance de ces variables de session sera vérifiée.
- En cas d'accès à une page avec des variables de session incorrectes, l'utilisateur sera redirigé automatiquement vers la page d'authentification

Organisation et démarches à suivre :

- ✓ Création de la table **users** aves ses quatre champs
- ✓ Création d'une page **register.php** qui permet l'inscription d'un nouvel utilisateur
- ✓ Création du fichier de connexion à la base des données « **config.php** »
- ✓ Création d'un formulaire de connexion **login.php** (vérifier les entrées utilisateur, si elles sont similaires avec celles dans la base des données, l'utilisateur sera redirigé vers la page d'accueil ; sinon un message d'erreur s'affiche).
- ✓ Création de la page d'accueil **index.php** qui permet d'initialiser une session, d'afficher : « *Bienvenue USER, Voici votre tableau de bord* » avec la possibilité de se déconnecter (lien) et ce dernier va permettre de détruire la connexion en faisant appel à une autre page (Logout).
- ✓ Création de la page **logout.php** qui permet de détruire la session et de rediriger l'utilisateur vers la première page de connexion.

Création de la table et connexion à la base de données

1

```sql
CREATE TABLE `users` (
  `id` int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `username` varchar(100) NOT NULL,
  `email` varchar(100) NOT NULL,
  `password` varchar(100) NOT NULL);
```

Création de la table « users »

2

```php
<?php
// Informations d'identification
define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'root');
define('DB_PASSWORD', '');
define('DB_NAME', 'session');

// Connexion à la base de données MySQL
$conn = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);

// Vérifier la connexion
if($conn === false){
    die("ERREUR : Impossible de se connecter. " . mysqli_connect_error());
}
?>
```

Après avoir créé la table, nous devons créer un script PHP afin de nous connecter au serveur de base de données MySQL. Créons un fichier nommé « **config.php** » et mettons le code suivant à l'intérieur.

Formulaire d'inscription d'un nouveau utilisateur

3

```php
<form class="box" action="" method="post">
    <h1 class="box-title">S'inscrire</h1>
  <input type="text" class="box-input" name="username" placeholder="Nom d'utilisateur" required />
    <input type="text" class="box-input" name="email" placeholder="Email" required />
    <input type="password" class="box-input" name="password" placeholder="Mot de passe" required />
    <input type="submit" name="submit" value="S'inscrire" class="box-button" />
    <p class="box-register">Déjà inscrit? <a href="login.php">Connectez-vous ici</a></p>
</form>
```

4

```php
require('config.php');
if (isset($_REQUEST['username'], $_REQUEST['email'], $_REQUEST['password'])){
  $username = $_POST['username'];
  $username = mysqli_real_escape_string($conn, $username);
  $email = $_POST['email'];
  $email = mysqli_real_escape_string($conn, $email);
  $password = $_POST['password'];
  $password = mysqli_real_escape_string($conn, $password);
```

Récupération des informations utilisateur

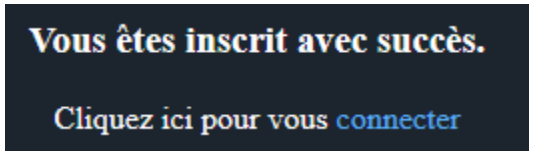
5

```php
$query = "INSERT into `users` (username, email, password)
          VALUES ('$username', '$email', '".hash('sha256', $password)."')";
 $res = mysqli_query($conn, $query);
```

Requête SQL + mot de passe crypté
Exécuter la requête sur la base de données

6

```php
if($res){
    echo "<div class='sucess'>
          <h3>Vous êtes inscrit avec succès.</h3>
          <p>Cliquez ici pour vous <a href='login.php'>connecter</a></p>
    </div>";
}
```



En cliquant sur connecter, on va être rediriger vers la page login.php

Création du formulaire login et connexion à l'interface si login OK

7

```php
<body>
<form class="box" action="" method="post" name="login">
<h1 class="box-title">Connexion</h1>
<input type="text" class="box-input" name="username" placeholder="Nom d'utilisateur">
<input type="password" class="box-input" name="password" placeholder="Mot de passe">
<input type="submit" value="Connexion " name="submit" class="box-button">
<p class="box-register">Vous êtes nouveau ici? <a href="register.php">S'inscrire</a></p>
```

8

```php
<?php
require('config.php');
session_start();
if (isset($_POST['username'])){
  $username = $_POST['username'];
  $username = mysqli_real_escape_string($conn, $username);
  $password = $_POST['password'];
  $password = mysqli_real_escape_string($conn, $password);
    $query = "SELECT * FROM `users` WHERE username='$username' and password='".hash('sha256', $password)."'";
  $result = mysqli_query($conn,$query) or die(mysql_error());
  $rows = mysqli_num_rows($result);
  if($rows==1){
      $_SESSION['username'] = $username;
      header("Location: index.php");
  }else{
    $message = "Le nom d'utilisateur ou le mot de passe est incorrect.";
  }
}
?>
```

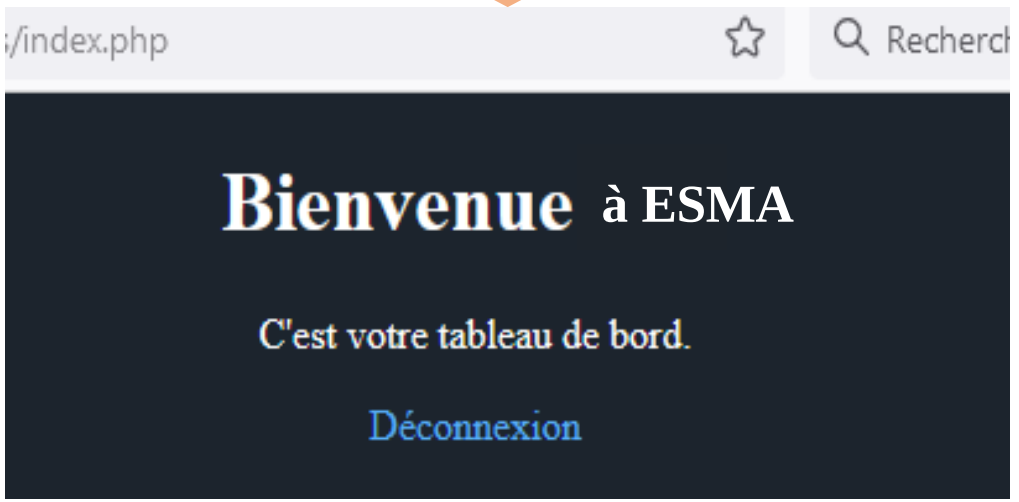
- Appel au fichier de connexion DB **config.php**.
- Ouverture d'une session
- Entrer des login **login.php** ou inscription pour nouveau user
- Vérification des identifiants avec ceux dans la DB.
- Si les entrées sont égaux à ceux dans la DB l'utilisateur sera rediriger vers la page **index.php, connexion OK**
- Si non, affichage d'un message d'erreur

Connexion à la page d'accueil si login OK

9

```php
<?php
  // Initialiser la session
  session_start();
?>
<!DOCTYPE html>
<html>
  <head>
  <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <div class="sucess">
    <h1>Bienvenue <?php echo $_SESSION['username']; ?>!</h1>
    <p>C'est votre tableau de bord.</p>
    <a href="logout.php">Déconnexion</a>
    </div>
  </body>
</html>
```

- La page login fait appel à la page **index.php** qui permet de:
- Initialisation de la session
- Affichage de la page principale
- Déconnexion possible via le fichier **logout.php**



10

```php
<?php
  // Initialiser la session
  session_start();
  // Détruire la session.
  if(session_destroy())
  {
    // Redirection vers la page de connexion
    header("Location: login.php");
  }
?>
```

En cliquant sur Déconnexion cela va détruire la session et nous rediriger vers la première page de connexion

Exercice_1

Vous allez créer une petite application de **gestion de véhicules**. Cette application doit permettre de :

- ✓ Définir une fonction pour calculer le nombre total de véhicules.
- ✓ Gérer une liste de véhicules à l'aide d'un tableau.
- ✓ Stocker et récupérer des informations sur les véhicules à l'aide des sessions.
- ✓ Gérer les préférences utilisateur à l'aide des cookies.

Étape 1 : Définir une fonction de calcul

Créez une fonction **somme** qui prend deux paramètres et retourne leur somme.

Utilisez cette fonction pour calculer le nombre total de véhicules disponibles et affichés.

Étape 2 : Gérer une liste de véhicules avec un tableau

- ✓ Créez un tableau contenant trois véhicules, chaque véhicule étant représenté par un tableau associatif avec les clés **marque** et **modele**.
- ✓ Affichez la liste des véhicules en utilisant une boucle **foreach**.

Exercice_1

Étape 3 : Stocker et récupérer des informations sur les véhicules à l'aide des sessions

- ✓ Démarrez une session et stockez le tableau des véhicules dans une variable de session.
- ✓ Affichez les informations des véhicules stockés dans la session.

Étape 4 : Gérer les préférences utilisateur à l'aide des cookies

- ✓ Créez un cookie pour stocker la préférence utilisateur pour le type de véhicule (par exemple, « BM", "Berline", etc.) avec une durée de vie de 30 jours.
- ✓ Vérifiez si le cookie est défini et affichez sa valeur. Sinon, définissez le cookie et affichez un message indiquant que la préférence a été enregistrée.
- ✓ Ajoutez une option pour supprimer le cookie en définissant sa durée de vie dans le passé et affichez un message indiquant que le cookie a été supprimé.

Exercice_2

Vous allez créer une petite application de gestion de produits pour un magasin en ligne. Cette application doit permettre de :

- Définir une fonction pour calculer le prix total des produits.
- Gérer une liste de produits à l'aide d'un tableau.
- Stocker et récupérer des informations sur les produits à l'aide des sessions.
- Gérer les préférences utilisateur (comme la devise préférée) à l'aide des cookies.

Étape 1 : Définir une fonction de calcul

- Créez une fonction **prixTotal** qui prend un tableau de produits et retourne le prix total de tous les produits.
- Chaque produit est représenté par un tableau associatif contenant les clés nom et prix.

Exercice_2

Étape 2 : Gérer une liste de produits avec un tableau

- Créez un tableau contenant trois produits, chaque produit étant représenté par un tableau associatif avec les clés nom et prix.
- Affichez la liste des produits en utilisant une boucle **foreach**.

Étape 3 : Stocker et récupérer des informations sur les produits à l'aide des sessions

- Démarrez une session et stockez le tableau des produits dans une variable de session.
- Affichez les informations des produits stockés dans la session.

Étape 4 : Gérer les préférences utilisateur à l'aide des cookies

- Créez un cookie pour stocker la devise préférée de l'utilisateur (par exemple, "EUR" ou "USD") avec une durée de vie de 30 jours.
- Vérifiez si le cookie est défini et affichez sa valeur. Sinon, définissez le cookie et affichez un message indiquant que la préférence a été enregistrée.

Exercice_2

- Ajoutez une option pour supprimer le cookie en définissant sa durée de vie dans le passé et affichez un message indiquant que le cookie a été supprimé.