

Technologie Web Dynamique et Mysql

LP - LDW - Temps aménagé

Outman El Hichami

ENS - Tétouan

2022/2023



المدرسة العليا للأساتذة
ECOLE NORMALE SUPERIEURE

Plan

1 Web dynamique

- Introduction
- PHP
 - Généralités
 - Variables
 - Chaînes de caractères
 - Tableaux
 - Structures de contrôle
 - Boucles
 - Fonctions
 - Formulaires

2 SGBD MySQL

- phpMyAdmin

1 Web dynamique

- Introduction
- PHP
 - Généralités
 - Variables
 - Chaînes de caractères
 - Tableaux
 - Structures de contrôle
 - Boucles
 - Fonctions
 - Formulaires

2 SGBD MySQL

- phpMyAdmin

Page web statique

Exemple

```
1 <!DOCTYPE html>
<html>
<head><title>Page statique</title></head>
<body>
Nous sommes le 17/01/2023
6 </body>
</html>
```

Problématique

Afficher une page différente selon l'utilisateur, l'environnement, ...

Solution

Utiliser un langage de programmation, par exemple **PHP**, **ASP**, **JSP**, ...



Page web statique

Exemple

```
<!DOCTYPE html>
<html>
3 <head><title>Page statique</title></head>
<body>
Nous sommes le 17/01/2023
</body>
</html>
```

Problématique

Afficher une page différente selon l'utilisateur, l'environnement, ...

Solution

Utiliser un langage de programmation, par exemple **PHP**, **ASP**, **JSP**, ...



Page web statique

Exemple

```
<!DOCTYPE html>
<html>
3 <head><title>Page statique</title></head>
<body>
Nous sommes le 17/01/2023
</body>
</html>
```

Problématique

Afficher une page différente selon l'utilisateur, l'environnement, ...

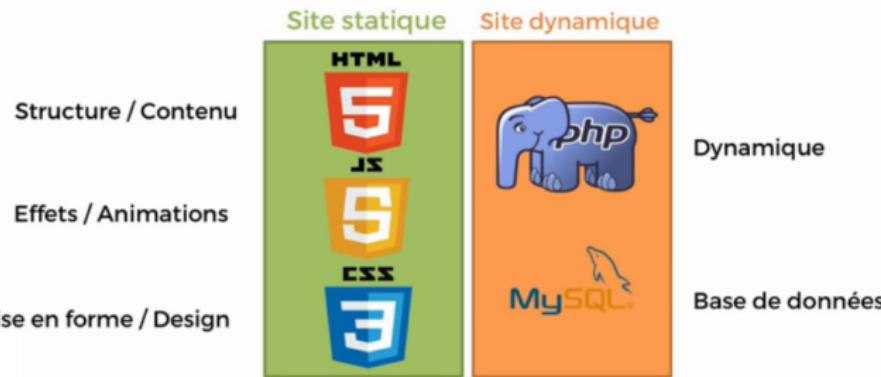
Solution

Utiliser un langage de programmation, par exemple **PHP**, ASP, JSP, ...



Introduction

Les langages



Introduction : Page web statique



Visiteur

Introduction : Page web statique



Introduction : Page web statique



Introduction : Page web dynamique

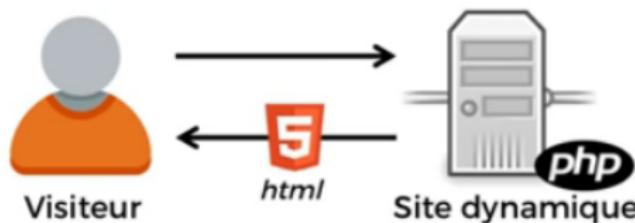


Visiteur

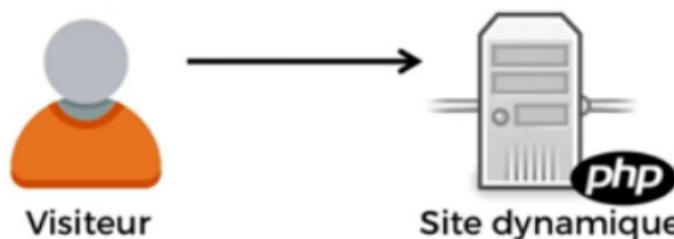
Introduction : Page web dynamique



Introduction : Page web dynamique



Introduction : Page web dynamique avec une base de données



Introduction : Page web dynamique avec une base de données



Introduction : Page web dynamique avec une base de données



Introduction : Page web dynamique avec une base de données



1 Web dynamique

- Introduction
- PHP
 - Généralités
 - Variables
 - Chaînes de caractères
 - Tableaux
 - Structures de contrôle
 - Boucles
 - Fonctions
 - Formulaires

2 SGBD MySQL

- phpMyAdmin

PHP

PHP

PHP : Hypertext Preprocessor (ou simplement **PHP**) est un langage de script côté serveur conçu pour le développement Web.

Que peut faire PHP ?

PHP peut :

- générer le contenu d'une page web dynamique
- créer, ouvrir, lire, écrire, supprimer et fermer des fichiers sur le serveur.
- collecter des données de formulaire
- ajouter, supprimer, modifier des données dans votre base de données
- être utilisé pour contrôler l'accès utilisateur
- chiffrer des données
- ...



PHP

PHP

PHP : Hypertext Preprocessor (ou simplement **PHP**) est un langage de script côté serveur conçu pour le développement Web.

Que peut faire PHP ?

PHP peut :

- générer le contenu d'une page web dynamique
- créer, ouvrir, lire, écrire, supprimer et fermer des fichiers sur le serveur.
- collecter des données de formulaire
- ajouter, supprimer, modifier des données dans votre base de données
- être utilisé pour contrôler l'accès utilisateur
- chiffrer des données
- ...



PHP

Configurer PHP sur votre PC

- installer un serveur web (**apache**)
- installer une base de données, telle que **MySQL**
- installer **PHP**

Installation de **WAMP**



Première page PHP

Exemple page statique

```
<!DOCTYPE html>
<html>
3 <head><title>Page statique</title></head>

<body>
Nous sommes le 17/01/2023
</body>
8
</html>
```

Première page PHP

Exemple page dynamique

```
1 <!DOCTYPE html>
<html>
<head><title>Page dynamique </title></head>

5 <body>
6 <?php
echo ( "Nous_sommes_le_<br>" ) ;
echo ( date ( "j-m-y" ) );
?>
</body>
11
</html>
```

Première page PHP

Exemple page dynamique (Résultat côté client)

```
<!DOCTYPE html>
<html>
3 <head><title>Page dynamique </title></head>

<body>
Nous sommes le 17/01/2023
</body>
8
</html>
```

Première page PHP

Exemple 3

```
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5      <h1>Ma Première page PHP</h1>
6
7      <?php
8          echo "Bonjour _ LDW _ Temps _ aménagé ";
9      ?>
10
11     </body>
12     </html>
```

PHP et HTML

Principe

- PHP s'intègre dans l'HTML entre `<?php` et `?>`
- Les instructions se finissent par `;`
- Les commentaires sont soit entre `/*` et `*/`, soit après `//` ou `#`
- Tutorial :
 - www.w3schools.com/php/,
 - www.php.net/manual/fr,
 - ...

1 Web dynamique

- Introduction
- PHP
 - Généralités
 - Variables
 - Chaînes de caractères
 - Tableaux
 - Structures de contrôle
 - Boucles
 - Fonctions
 - Formulaires

2 SGBD MySQL

- phpMyAdmin

Variables

Principe

- Les variables sont prefixées par \$
- Leur nom suit les règles classiques

Exemple : \$une_var1

- Les noms sont sensibles à la casse : \$ldw1 ≠ \$Ldw1
- Pas de déclaration, typage implicite

Exemple :

```
$une_var1 = 2023 ; // Maintenant, c est un  
                    entier  
$une_var1 = "LDW" ; // Maintenant, c est une  
                    chaine
```

Variables : Affectation

- Tester ces deux exemples :

```
<?php
$var1 = 1;
3 $var2 = $var1; //Affectation par valeur
$var1 = "LDW";
echo $var1, $var2;
?>19
```

```
<?php
$var1 = 1;
$var2 = &$var1; //Affectation par référence
4 $var1 = "LDW";
echo $var1, $var2;
?>
```

Que ce que vous remarquez ?

Variables : Types

Types

- Entiers : 2023
- Float : 3.19
- Chaines : "LDW" ou 'LDW'
- Booléens : false ou true
- Tableaux (Array)

Variables : gettype

Pour déterminer le type d'une variable, tester ces exemples :

```
<?php  
$var=2023;  
echo gettype($var); // integer
```

4

```
$var="ENS";  
echo gettype($var); // string
```

9

```
$var=16.5;  
echo gettype($var); // double
```

```
$var=TRUE;  
echo gettype($var); // boolean
```

14

```
$var=NULL;  
echo gettype($var); // NULL
```

ENS - Tétouan 28/86



Variables

Pour vérifier si une variable est d'un type précis :

```
<?php  
is_integer($var) //ou is_int($var)  
is_int($var)  
4 is_double($var)  
is_bool($var)  
is_array($var)  
is_null($var)  
  
9 /* Ces fonctions retournent la valeur booléenne  
     TRUE si la variable est du type recherché et  
     FALSE dans le cas contraire */  
?>
```

Variables : Etat d'une variable

Contrôler l'état d'une variable

- **isset(\$var)** : tester si une variable \$var existe
- **empty(\$var)** : tester si une variable \$var est nulle (**Très utile pour vérifier l'état des données envoyées par un formulaire**)

Conversion de type

- Cas de données issues de formulaire : ces variables sont toujours de type **string**
- Très utile de convertir explicitement une variable d'un type à un autre
- Deux méthodes :

```
$result = (type_désiré) $var;  
settype($var, "type_désiré");
```

Variables : Etat d'une variable

Contrôler l'état d'une variable

- **isset(\$var)** : tester si une variable \$var existe
- **empty(\$var)** : tester si une variable \$var est nulle (**Très utile pour vérifier l'état des données envoyées par un formulaire**)

Conversion de type

- Cas de données issues de formulaire : ces variables sont toujours de type **string**
- Très utile de convertir explicitement une variable d'un type à un autre
- Deux méthodes :

```
$result = (type_désiré) $var;  
settype($var, "type_désiré");
```

Variables : Etat d'une variable

Contrôler l'état d'une variable

- **isset(\$var)** : tester si une variable \$var existe
- **empty(\$var)** : tester si une variable \$var est nulle (**Très utile pour vérifier l'état des données envoyées par un formulaire**)

Conversion de type

- Cas de données issues de formulaire : ces variables sont toujours de type **string**
- Très utile de convertir explicitement une variable d'un type à un autre
- Deux méthodes :

```
$result = (type_désiré) $var;  
settype($var, "type_désiré");
```

Variables : Etat d'une variable

Contrôler l'état d'une variable

- **isset(\$var)** : tester si une variable \$var existe
- **empty(\$var)** : tester si une variable \$var est nulle (**Très utile pour vérifier l'état des données envoyées par un formulaire**)

Conversion de type

- Cas de données issues de formulaire : ces variables sont toujours de type **string**
- Très utile de convertir explicitement une variable d'un type à un autre
- Deux méthodes :

```
$result = (type_désiré) $var;  
settype($var, "type_désiré");
```

Variables : Etat d'une variable

Tester ces exemples

```
<?php  
$var="3.52 kilomètres";  
$var2 = (double) $var;  
echo "\$var2=_",$var2,"<br>"; //affiche "$var2  
=3.52"  
  
$var3 = (integer) $var2;  
echo "\$var3=_",$var3,"<br>"; //affiche "$var3=3"  
$var4 = (boolean) $var3;  
echo "\$var4=_",$var4,"<br>"; //affiche "$var4=1"  
soit la valeur true  
?>
```

Variables : Etat d'une variable

Tester ces exemples

```
<?php  
$var = "5_Max"; // chaîne  
$bol = true; // booléen  
4  
settype($var, "integer"); // $var vaut maintenant 5  
    (integer)  
settype($bol, "string"); // $bol vaut maintenant  
    "1" (string)  
  
echo "\$var = ", $var, "<br>\$bol = ", $bol;  
9 ?>
```

Variables : Exercice

- Donnez les valeurs de \$x, \$y, \$z à la fin du script suivant :

```
1 $x="NoSQL";
  $y="MySQL";
  $z=&$x;
  $x="PHP_7";
  $y=&$x;
6 echo "\$x=", $x, "\$y=", $y, "\$z=", $z;
```

- Donnez les valeurs de \$x, \$y, \$z à la fin du script suivant :

```
$x="7 personnes";
$y=(integer) $x;
$x="9E3";
4 $z=(float) $x;
echo "\$x=", $x, "\$y=", $y, "\$z=", $z;
```

Variables : chaînes de caractères

Qu'affiche le script suivant :

```
$var= "ENS";
echo 'Bonjour_' . $var;
echo "Bonjour_" . $var;
echo "Bonjour_$var";
5 echo 'Bonjour_$var';

print("Bonjour_$var_");
print("Bonjour_" . $var);
print("Bonjour_"). $var;
10 print('Bonjour_'. $var);
print('Bonjour_'). $var;
print('Bonjour_..$var');
```

Variables : chaînes de caractères

Conversion de chaînes de caractères en valeur numérique

La valeur d'une chaîne est définie par la première partie de la chaîne (0 si c'est du texte)

Qu'affiche le script suivant :

```
$i=1+ "4.5"; // i = 5.5
echo $i;
3 $i=1+ "-1e3"; // i = -999
echo $i;
$i=1+ "chaine_9"; // i = 1
echo $i;
$i=1+ "9_chaine"; // i = 10
8 echo $i;
```

Variables : chaînes de caractères

Accès aux caractères d'une chaîne :

Qu'affiche le script suivant :

```
$ch = "Ceci_est_une_chaine";  
echo $ch{5}; // affiche la lettre e
```

Constante

Deux méthodes pour utiliser les constantes : (pas de signe \$)

```
define("COULEUR", "rouge");
Const PI= 3.14;
3 echo COULEUR , '<br>';
echo PI;
```

Opérateurs

Opérateurs arithmétiques

- + : addition
- - : soustraction
- / : division
- * : multiplication
- % : modulo
- ++ : incrémentation
- -- : décrémentation

Opérateurs

Opérateurs d'affectation

- `=` : affectation
- `+=` : addition puis affectation
- `-=` : soustraction puis affectation
- `*=` : multiplication puis affectation
- `/=` : division puis affectation
- `%=` : modulo puis affectation

Exercices : Test des exemples d'affectation

Opérateurs

Opérateurs de comparaison

- `==` : égalité
- `>` : inférieur strictement
- `<` : supérieur strictement
- `<=` : inférieur ou égale
- `>=` : supérieur ou égal
- `!=` : négation

Opérateurs

Opérateurs logiques

- `&&` : et
- `||` : ou
- `xor` : ou exclusif
- `!` : négation

Les opérateurs **and**, **or** et **not** sont également disponibles et font la même chose.

Opérateurs

Opérateurs divers

- L'opérateur de **concaténation** :

```
1 $chaîne = "LDW1" ;
  $nom = "LDW2";
  echo $chaîne . "_et_" . $nom;
```

- L'opérateur **? : [test logique] ? [expression si vrai] : [expression si faux]**

```
$a = $b = 1;
2  ($a == $b) ? $c = 10 : $c = 20;
echo $c
```

```
$réponse = ($a == $b) ? "a_égal_b" : "a_diffé
rent_de_b";
echo $réponse;
```

Opérateurs

Opérateurs divers

- L'opérateur de **concaténation** :

```
$chaîne = "LDW1" ;  
$nom = "LDW2" ;  
echo $chaîne . "_et_" . $nom;
```

- L'opérateur **? : [test logique] ? [expression si vrai] : [expression si faux]**

```
$a = $b = 1;  
( $a == $b ) ? $c = 10 : $c = 20;  
echo $c
```

```
$réponse = ($a == $b) ? "a_égal_b" : "a_diffé  
rent_de_b" ;  
echo $réponse;
```



Opérateurs

Opérateurs divers

- L'opérateur de **concaténation** :

```
$chaîne = "LDW1" ;
$nom = "LDW2";
echo $chaîne . "_et_" . $nom;
```

- L'opérateur **? : [test logique] ? [expression si vrai] : [expression si faux]**

```
$a = $b = 1;
($a == $b) ? $c = 10 : $c = 20;
echo $c
```

```
$réponse = ($a == $b) ? "a_égal_b" : "a_diffé
rent_de_b" ;
echo $réponse;
```

Tableaux

- **Syntaxe :**

```
$classes= array(); //Pour créer un tableau vide
```

- **Affectation :**

```
//Affectation élément par élément  
$classes[0]= 'lpr1';  
$classes[1]= 'lpr2';  
$classes[] = 'ipm1';//Ou $classes[2]= 'ipm1'  
//Affectation multiple  
$classes= array( 'lpr1', 'lpr2', 'ipm1' );
```

- **Affichage :**

```
echo $classes[1]. '<br>'; //Affiche la case 1  
print_r($classes); //Affiche la structure complète du tableau
```



Tableaux

- **Syntaxe :**

```
$classes= array(); //Pour créer un tableau vide
```

- **Affectation :**

```
//Affectation élément par élément  
$classes[0]= 'lpr1';  
$classes[1]= 'lpr2';  
$classes[] = 'ipm1';//Ou $classes[2]= 'ipm1'  
//Affectation multiple  
$classes= array( 'lpr1', 'lpr2', 'ipm1' );
```

- **Affichage :**

```
echo $classes[1]. '<br>'; //Affiche la case 1  
print_r($classes); //Affiche la structure complète du tableau
```



Tableaux

- **Syntaxe :**

```
$classes= array(); //Pour créer un tableau vide
```

- **Affectation :**

```
//Affectation élément par élément  
$classes[0]= 'lpr1';  
$classes[1]= 'lpr2';  
$classes[] = 'ipm1';//Ou $classes[2]= 'ipm1'  
//Affectation multiple  
$classes= array( 'lpr1', 'lpr2', 'ipm1' );
```

- **Affichage :**

```
echo $classes[1]. '<br>'; //Affiche la case 1  
print_r($classes); //Affiche la structure complète du tableau
```



Tableaux

Parcourir les tableaux

```
$classes= array( 'lpr1', 'lpr2', 'ipm1' );  
  
3 for( $i=0; $i < count($classes); $i++ ) {  
    echo "La classe $i = $classes[$i]<br>";  
}  
  
foreach($classes as $cls){  
8    echo "$cls<br>";  
}
```

Tableaux

Parcourir les tableaux

```
1 $i=0;
  while( iset($classes[$i]) ) {
    echo "La classe $i = $classes[$i]<br>";
    $i++;
}
6
  $i=0;
  do{
    echo "La classe $i = $classes[$i]<br>";
    $i++;
11 }while(iset($classes[$i]))
```

Tableaux

Fonctions pour les tableaux

- **count** ou bien **sizeof** : Renvoie le nombre d'éléments d'un tableau
- **is_array** : retourne **true** si la variable est de type tableau, **false** sinon.
- **echo reset(\$classes)** : Affiche le premier élément du tableau
- **echo end(\$classes)** : Affiche le dernier élément du tableau.
- **echo current(\$classes)** : Affiche l'élément courant du tableau.
- **echo next(\$classes)** : Affiche l'élément suivant. S'il n'existe pas, la fonction retourne **false**.
- **echo prev(\$classes)** : Affiche l'élément précédent. S'il n'existe pas, la fonction retourne **false**.
- **sort** : trie le tableau par valeurs croissantes.
- **rsort** : trie le tableau par valeurs décroissantes.
- **array_push** : Insère un ou plusieurs éléments à la fin d'un tableau
- **array_pad** : insère un nombre spécifié d'éléments dans un tableau
- **array_shift** : supprime le premier élément d'un tableau et retourne la valeur supprimée
- **array_pop** : supprime le dernier élément
- ...

Tableau associatif

Définition

Un **tableau associatif** est un tableau dont l'index est, peut être, une chaîne de caractère ou bien un nombre entier ou réel.

Il existe deux façons de créer un tableau associatif :

```
$classes= array(); //Tableau associatif vide  
  
//Méthode 1  
4 $Infos = array('Mohamed'=>18, 'Fatima'=>17, 1000=>  
    '1kg');  
  
//Méthode 2  
$Infos['Mohamed'] = 18;  
$Infos['Fatima'] = 17;  
9 $Infos[1000] = '1kg';
```



Tableau associatif

Parcourir les tableaux associatifs

```
1 $Infos = array('Mohamed'=>18, 'Fatima'=>17, 1000=>'  
    1kg');  
  
foreach($Infos as $cle => $valeur){  
    echo "$cle : $valeur  
";  
}
```

Fonctions pour les tableaux associatifs

- **asort** : Trie par ordre croissant, en fonction de la valeur
- **arsort** : Trie par ordre décroissant, en fonction de la valeur
- **ksort** : Trie par ordre croissant, en fonction de la clé
- **krsort** : Trie par ordre décroissant, en fonction de la clé



Tableaux à deux dimensions

Définition

Nom	Note 1	Note 2
Mohamed	13	15
Fatima	15	16
khadija	15	19
Farid	1	3

Ces données peuvent être stockées dans un tableau à deux dimensions :

```
$Notes = array  
(  
    array('Mohamed', 13, 15),  
    array('Fatima', 15, 16),  
5    array('khadija', 15, 19),  
    array("Farid", 1, 3)  
) ;
```



Tableaux à deux dimensions

Définition

Nom	Note 1	Note 2
Mohamed	13	15
Fatima	15	16
khadija	15	19
Farid	1	3

Ces données peuvent être stockées dans un tableau à deux dimensions :

```
$Notes = array  
(  
    array('Mohamed', 13, 15),  
    array('Fatima' ,15,16),  
    array('khadija',15,19),  
    array("Farid",1,3)  
)
```



Tableaux à deux dimensions

Parcourir un tableau à deux dimensions

```
$Notes = array
(
    array('Mohamed', 13, 15),
    array('Fatima', 15, 16),
    array('khadija', 15, 19),
    array("Farid", 1, 3)
);
echo "La_ligne_0_:_". $Notes[0][0]. "_". $Notes[0][1].
      "_". $Notes[0][2]. "<br>";
echo "La_ligne_1_:_". $Notes[1][0]. "_". $Notes[1][1].
      "_". $Notes[1][2]. "<br>";
echo "La_ligne_2_:_". $Notes[2][0]. "_". $Notes[2][1].
      "_". $Notes[2][2]. "<br>";
echo "La_ligne_3_:_". $Notes[3][0]. "_". $Notes[3][1].
```

Tableaux à deux dimensions

Parcourir un tableau à deux dimensions

```
$Notes = array
(
    array('Mohamed', 13, 15),
    array('Fatima', 15, 16),
    array('khadija', 15, 19),
    array("Farid", 1, 3)
);
for ($i=0; $i < count($Notes) ; $i++) {
    echo "La_ligne_$i:_";
    for ($j=0; $j < count($Notes[0]); $j++)
        echo $Notes[$i][$j]. '_';
    echo '<br>';
}
```

Tableaux à deux dimensions

Exercices

- ① Définir et afficher deux matrices **M1** et **M2**.

- ② Ecrire une fonction **matrice_somme**(M1, M2) qui calcule et affiche, **si c'est possible**, la matrice somme **S** de deux matrices M1 et M2, sinon il affiche un **message d'erreur**.

- ③ Ecrire une fonction **matrice_produit**(M1, M2) qui calcule et affiche, **si c'est possible**, la matrice produit **P** de deux matrices M1 et M2, sinon il affiche un **message d'erreur**.