esprit
Se former autrement

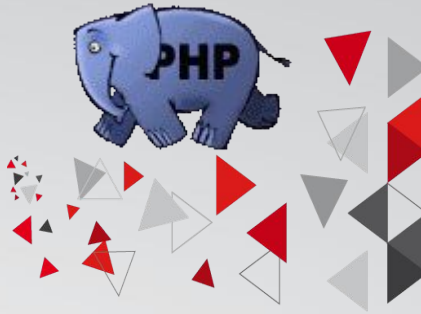
PHP

Hypertext Preprocessor Présentation

Année universitaire
2017-2018

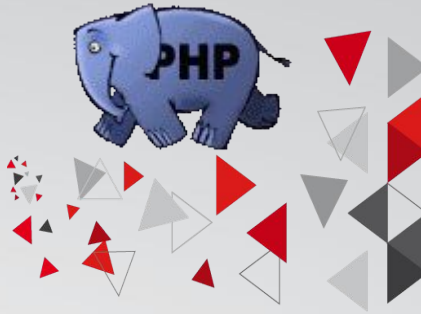
United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
CONFÉRENCE DES GRANDES ÉCOLES
EXCELLENCE FOR A COMPLEX WORLD
European Accreditation of Engineering Programmes EUR-ACE®
délivrée par CTI Commission des Titres d'Ingénieur
cdio
CONCEIVE DESIGN IMPLEMENT OPERATE

Plan



- C’est quoi PHP?
- Evolution de PHP
- Fonctionnement de PHP
- Environnement de travail
- Eléments de base du langage PHP
    - Structure de base d’une page PHP
    - Les variables
    - Les types

Plan



Les tableaux

Les constantes

Les opérateurs

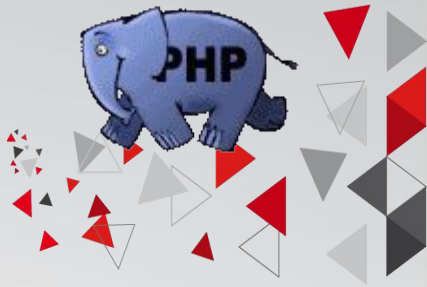
Les structures conditionnelles

Les boucles

Déclaration de fonction

Les fonctions prédéfinies utiles

C’est quoi PHP?



- un **langage de programmation** qui s'intègre dans les pages HTML
- C’est **un langage de scripts** Open Source
- Conçu pour le développement d'**applications web dynamiques**
- Un langage de script dynamique précompilé et **interprêté côté serveur**

Évolution de PHP

PHP/F1
1995

PHP/F2
1997

PHP 3.0
1998

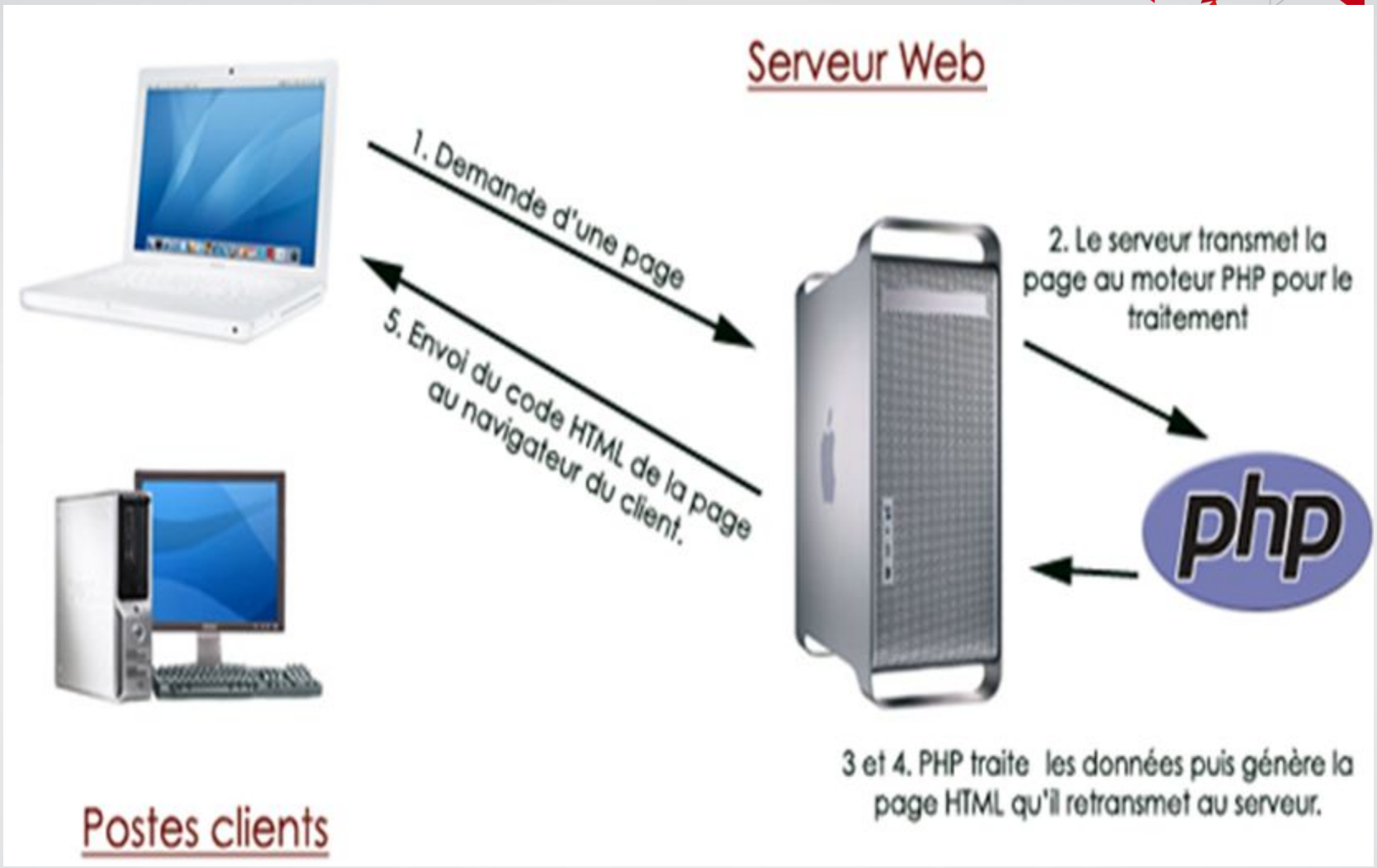
PHP 4
2000

PHP 5
2004

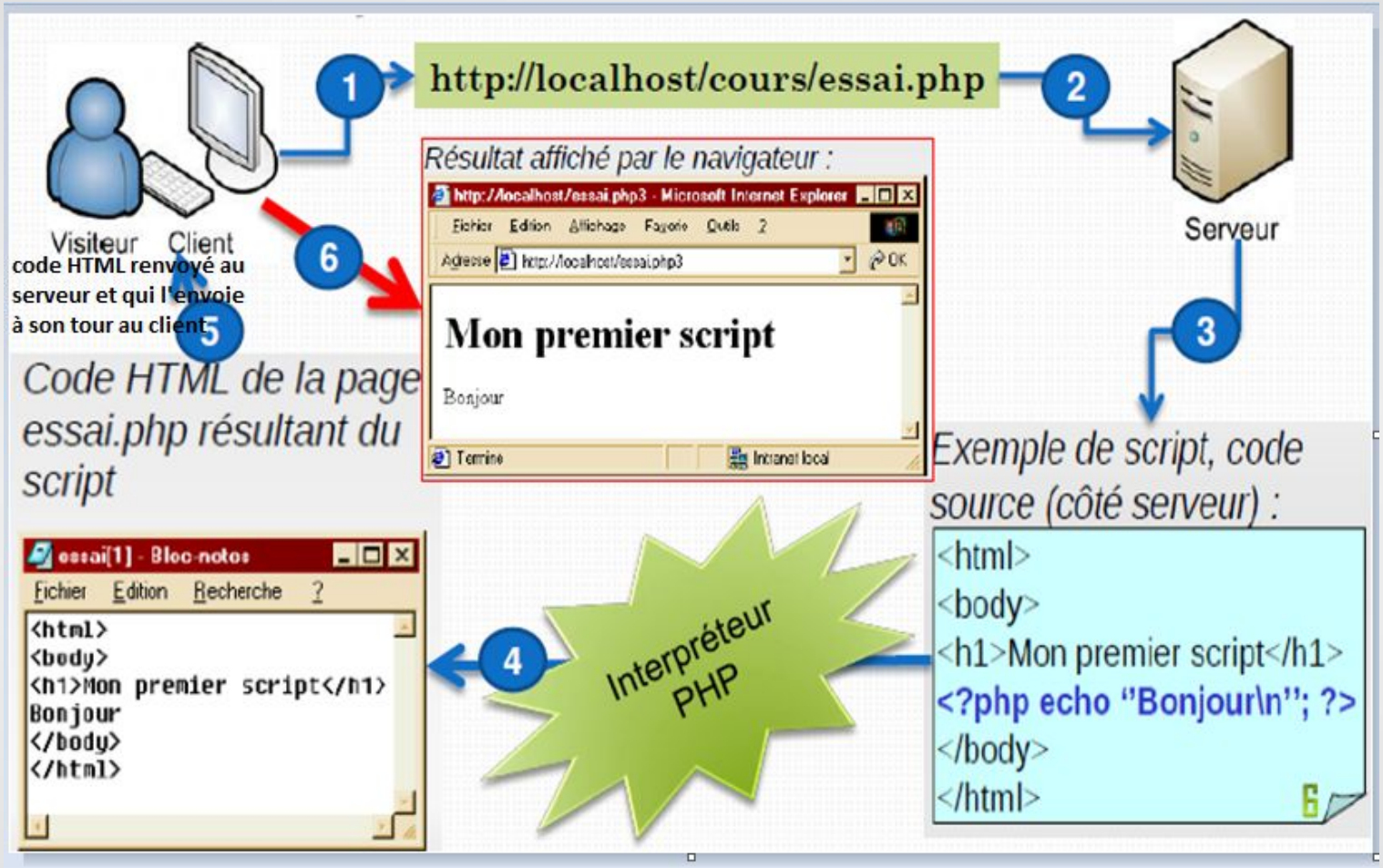
PHP 7
2015

esprit
Se former autrement

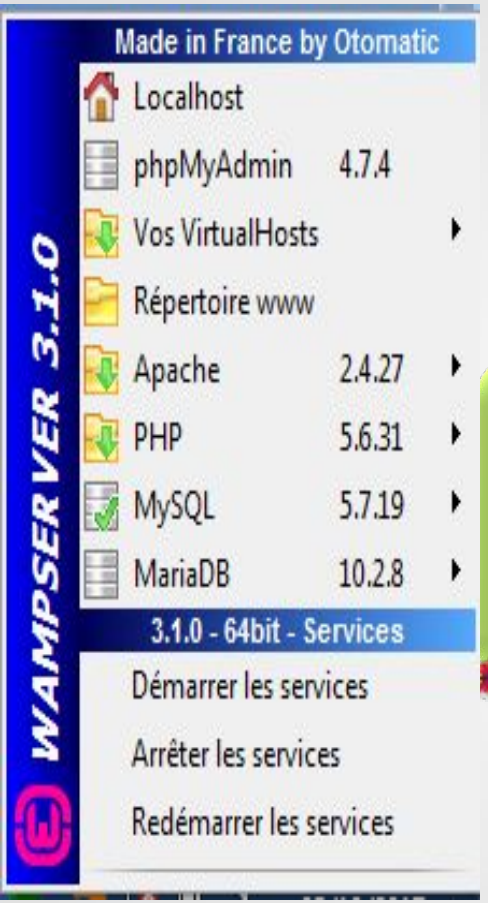
Fonctionnement de PHP (1/2)



Fonctionnement de PHP (2/2)



Environnement de travail Wampserver



MySQL est un système de gestion de bases de données relationnelles (SGBDR)

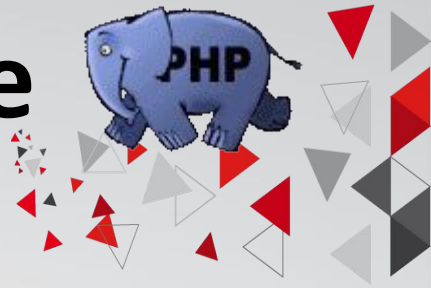
Localhost: Lancement du serveur local

Apache est Le logiciel libre **Apache HTTP Server** (**Apache**) est un serveur HTTP créé et maintenu au sein de la fondation Apache. C'est le serveur HTTP le plus populaire du World Wide Web.

www directory: Répertoire de base du serveur: la racine du serveur

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes d gestion de base de données MySQL réalisée en PHP et distribuée sous licence GNU GPL

Structure de base d'une page PHP



Les instructions PHP sont placées entre les balises:

```
<?php
// Le texte du script
?>
```

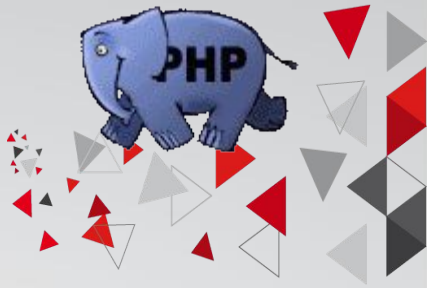
Page d'extension .php

Commentaires : // (une seule ligne) ou bien /*….*/ (plusieurs lignes)

```
<?php
/* Commentaire sur 2
lignes */
?>
```

Chaque instruction se termine par ;

Les variables (1/2)

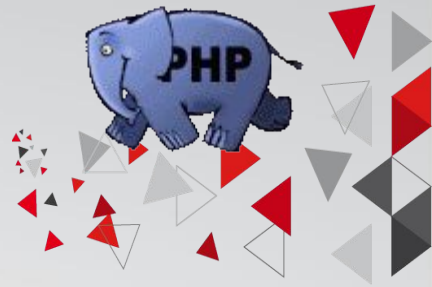


- Une variable **doit** commencer par le signe **$**.
- PHP est sensible à la casse (majuscule et minuscule), mavar et MaVar ne désignent pas la même chose.
- La déclaration des variables n'est pas obligatoire en début de script.
- Pour afficher le contenu de la variable on utilise le mot clé **echo**

```php
<?php
$toto='<p>Hello world!</p>';
echo $toto;
?>
```

Résultat:
Hello world!

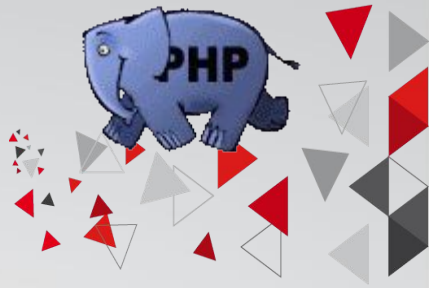
Les variables (2/2)




```php
<?php
    $toto_fr='<p>Hello world!</p>';
    $toto_langue='toto_fr';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    echo $$toto_langue;
    ?>
</body>
</html>
```

Résultat:
Hello world!

Les types (1/6)

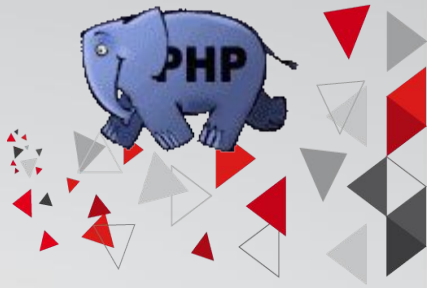


- Avec PHP nous pouvons utiliser la même variable pour stocker et afficher les différents types de données.
- La variable change de type en fonction du contenu qu'elle reçoit.

→ Le type des variables est dynamique, léger et souple

→ PHP est un langage à typage faible et dynamique.

Les types (2/6)



Exemple:

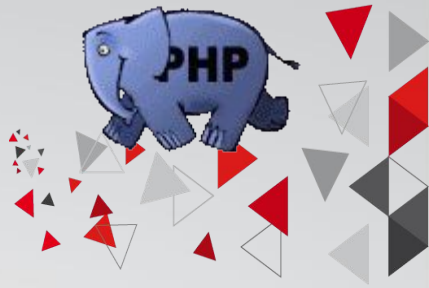
```
$mavariable=123;                    //entier

$mavariable=false;                  //booléen

$mavariable=3.14159;                //flottant

$mavariable='bonjour ';             //chaîne
```

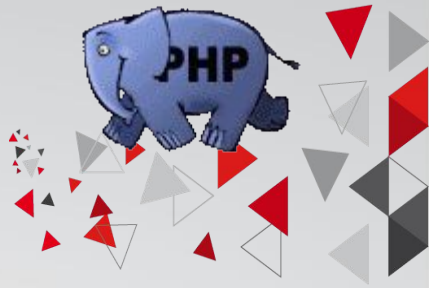
Les types (3/6)



Il existe deux types:

- **Les types scalaires**: ce sont les types qui pouvaient être affichés directement par le navigateur (booléen, entier, décimal, chaine de caractère).
- **Les types complexes**: les tableaux (array), les objets, …

Les types (4/6)

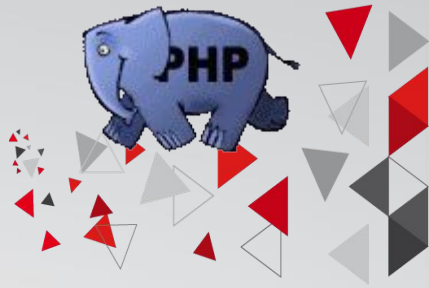


Les types scalaires

```
<?php
     $retour='<br/>';
     ?>
<!DOCTYPE html>
<html>
<head>
     <title>Document PHP</title>
</head>
<body>
     <?php
     $toto=1;
     echo $toto;
     echo $retour;
     $toto=1.5;
     echo $toto;
     echo $retour;
     $toto='Bonjour';
     echo $toto;
     ?>
</body>
</html>
```

Résultat:
1
1.5
Bonjour

Les types (5/6)

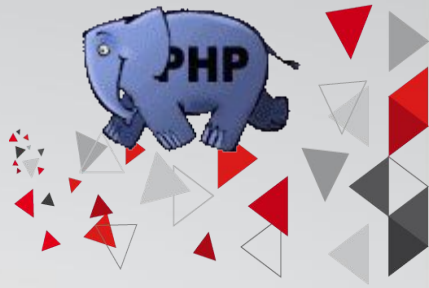


Les types complexes

```php
<?php
    $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    $toto=array('tab1','tab2');
    echo $toto[0];
    echo $retour;
    echo $toto[1];
    ?>
</body>
</html>
```

Résultat :
tab1
tab2

Les types (6/6)

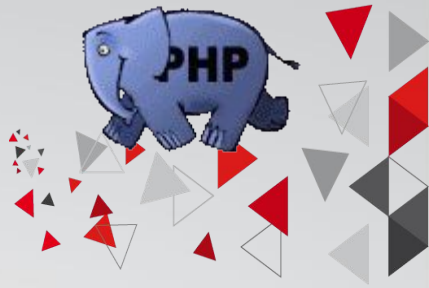


Les types complexes

```php
<?php
    $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    $toto=array('tab1','tab2');
    echo $toto[0];
    echo $retour;
    echo $toto[1];
    ?>
</body>
</html>
```

Résultat:
tab1
tab2

Les tableaux (1/7)

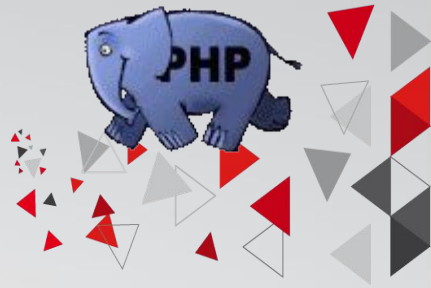


Il existe deux types de tableaux:

- **Les tableaux indexés**: Chaque élément du tableau est identifié par son index (0,1,2,..)

- **Les tableaux associatifs**: On associe à chaque élément du tableau une clé dont la valeur est de type chaine de caractères

Les tableaux (2/7)



Les tableaux indexés : déclaration

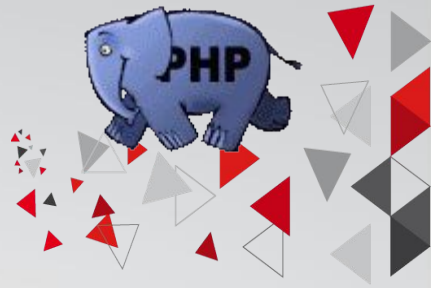
```
<?PHP
$T=array(2,3,6);
print_r($T);
?>
```

```
<?PHP
$T=array();
$T[]=2;
$T[]=3;
 $T[]=6;
print_r($T);
?>
```

```
<?PHP
$T[0]=2;
$T[1]=3;
$T[2]=6;
print_r($T);
?>
```

Résultat: Array ([0]=>2 [1]=>3 [2]=>6)

Les tableaux (3/7)

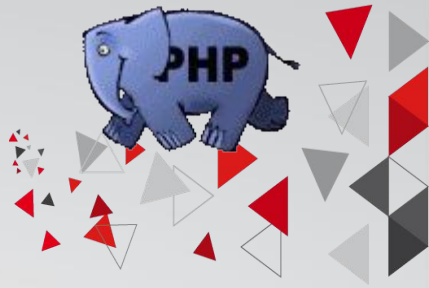



```php
<?php
    $tableau=array('<p>Hello</p>','<p>Bonjour</p>');
    $affiche=0;
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    echo $tableau[$affiche];
    ?>
</body>
</html>
```

Résultat:

Hello

Les tableaux (4/7)



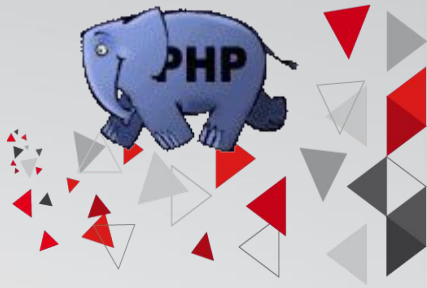
Les tableaux associatifs: déclaration

```php
<?PHP
$personne = array('nom'=>'Ali',
'Prenom'=>'Salah');
print_r($personne);
?>
```

```php
<?PHP
$personne['nom']='Ali';
$personne['prenom']='Salah';
print_r($personne);
?>
```

Résultat: Array ([nom] => Ali [Prenom] => Salah)

Les tableaux (5/7)

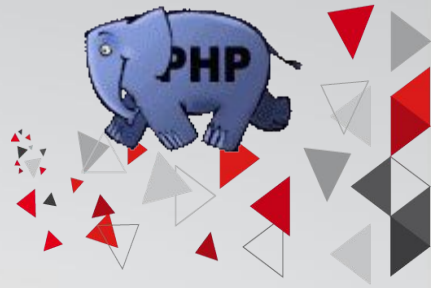


RQ:
Les clés des tableaux associatifs sont sensibles à la casse.

$T[‘cle’] est différent de $T[‘CLE’]

Les chaînes définissants les clés ne doivent pas contenir des espaces

Les tableaux (6/7)

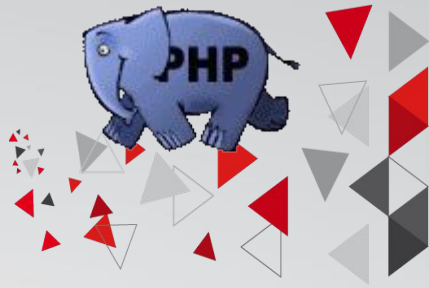



```php
<?php
    $tableau=array();
    $tableau['ch1']='<p>Hello</p>';
    $tableau['ch2']='<p>Bonjour</p>';
    $affiche='ch2';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    echo $tableau[$affiche];
    ?>
</body>
</html>
```

Résultat:

Bonjour

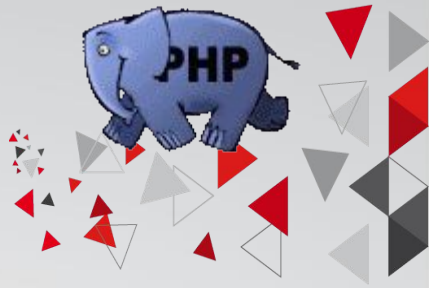
Les tableaux (7/7)



Fonction prédéfinies sur les tableaux:

Fonction	Explication
count($T), sizeof($T)	Retourne le nombre des éléments du tableau **$T**
in_array($var,$T)	Teste si la valeur de **$var** existe dans le tableau **$T**
sort($T)	trie alphanumérique les éléments du tableau et réaffecte les indices du tableau
…	

Les constantes (1/2)



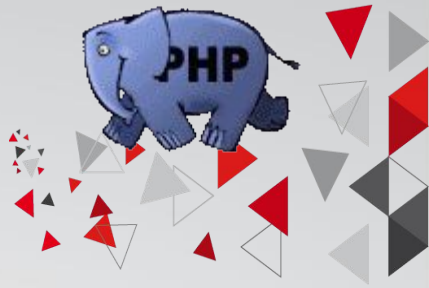
Une constante est une valeur qui ne pourra plus être modifiée.

```php
<?php
    define('AFFICHE','Hello');
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    echo AFFICHE;
    ?>
</body>
</html>
```

Résultat:

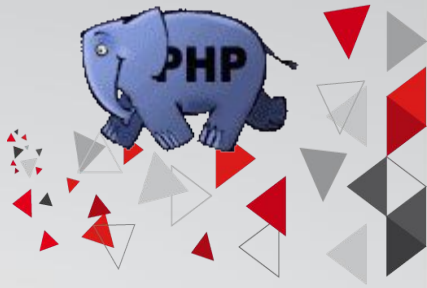
Hello

Les constantes (2/2)



- Par convention le nom de la constante doit être en majuscule.
- Pour définir une constante, on n'utilise pas le **$**.
- La fonction define retourne TRUE en cas de succès et FALSE en cas de problème

Les opérateurs (1/5)

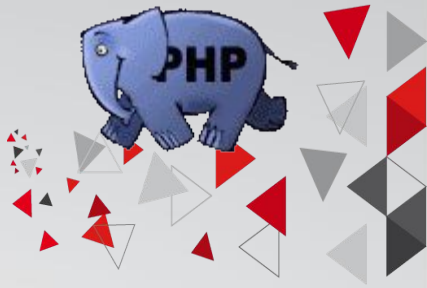


Les opérateurs servent à comparer, à calculer et à combiner des valeurs.

Les principaux opérateurs sont:

- Les opérateurs de concaténation: « . »
- Les opérateurs arithmétiques: « +, -, *, / »
- Les opérateurs d'affectation.
- Les opérateurs de comparaison.
- Les opérateurs logiques.

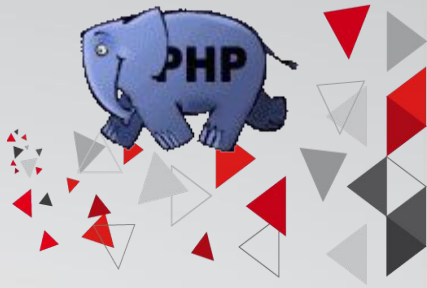
Les opérateurs (2/5)



Les opérateurs d'affectation

Exemple	Résultat
$a=1; $b=$a; echo $b;	1
$a=1; echo $a+=1;	2
$a=1; echo $a-=1;	0
$a=1; echo $a*=1;	1
$a=1; echo $a/=1;	1

Les opérateurs (3/5)

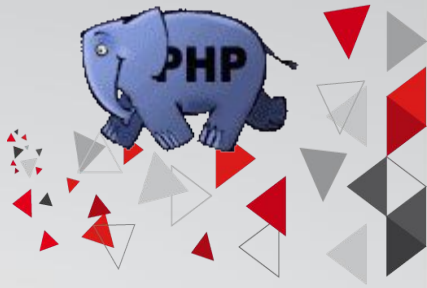


Les opérateurs d'affectation

Exemple	Résultat
$a='Hello'; $b=' ahmed'; echo $a.=$b;	Hello ahmed
$a=1; $a++; echo $a;	2
$a=1; ++$a; echo $a;	2

$a=1;
echo $a++;
echo '
';
echo $a;
echo '
';
echo ++$a;
echo '
';
echo $a;
	1
	2
	3
	3

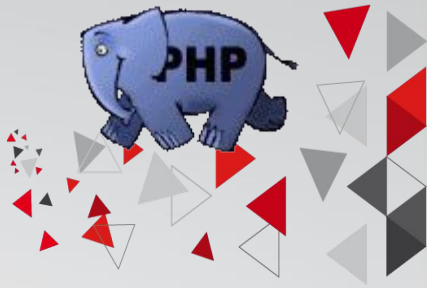
Les opérateurs (4/5)



Les opérateurs de comparaison

Opérateur	Valeur
==	égal
===	identique.
<>	non égal
!=	non égal
!= =	non identique
<	inférieur
>	supérieur
<=	inférieur ou égal
>=	supérieur ou égal

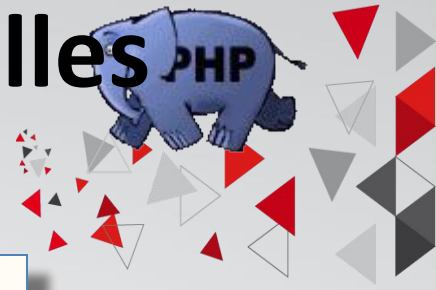
Les opérateurs (5/5)



Les opérateurs de comparaison

Opérateur	Valeur
&a and &b	True si $a est true et $b est true
&a or &b	True si $a est true ou $b est true
&a xor &b	True si $a est ou $b est true mais non pas les deux à la fois
! &a	True si $a n'est pas true
&a && &b	True si $a est true et $b est true
&a || &b	True si $a est true ou $b sont true

Les structures conditionnelles (1/3)



if else

```php
<?php
      define('AFFICHE','ch2');
      $ch1='<p>Hello</p>';
      $ch2='<p>Bonjour</p>';
      $ch3='<p>Salut</p>';
      ?>
<!DOCTYPE html>
<html>
<head>
      <title>Document PHP</title>
</head>
<body>
      <?php
      if(AFFICHE=='ch1')
      {
            echo $ch1;
      }
      else if(AFFICHE=='ch2')
      {
            echo $ch2;
      }
      else
      {
            echo $ch3;
      }
      ?>
</body>
</html>
```

Résultat:

Bonjour

Les structures conditionnelles (2/3)

switch (EXP1)

```php
<?php
    define('AFFICHE','ch3');
    $ch1='<p>Hello</p>';
    $ch2='<p>Bonjour</p>';
    $ch3='<p>Salut</p>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    switch(AFFICHE)
    {
        case'ch1':
            echo $ch1;
            break;
        case'ch2':
            echo $ch2;
            break;
        case'ch3':
            echo $ch3;
            break;
        default:
            echo 'votre chaine n\'existe pas';
    }
    ?>
</body>
</html>
```

Résultat:

Salut

Les structures conditionnelles (3/3)

switch (EXP2)

```php
<?php
    define('AFFICHE','ch3');
    $ch1='<p>Hello</p>';
    $ch2='<p>Bonjour</p>';
    $ch3='<p>Salut</p>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    switch(AFFICHE):
        case'ch1':
            echo $ch1;
            break;
        case'ch2':
            echo $ch2;
            break;
        case'ch3':
            echo $ch3;
            break;
        default:
            echo 'votre chaine n\'existe pas';
    endswitch;
    ?>
</body>
</html>
```

Résultat:
Salut

Les boucles

Boucle for

```php
<?php
        $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    for($i=0;$i<=3;$i++)
    {
        echo $i.$retour;
    }
    ?>
</body>
</html>
```

Résultat:

0
1
2
3

Les boucles

Boucle while

```php
<?php
        $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    $i=0;
    while($i<=3)
    {
        echo $i.$retour;
        $i++;
    }
    ?>
</body>
</html>
```

Résultat:

0

1

2

3

esprit
Se former autrement

Les boucles

Boucle do..while

```php
<?php
        $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    $i=0;
    do
    {
        echo $i.$retour;
        $i++;
    }
    while($i<=3);
    ?>
</body>
</html>
```

Résultat:

0
1
2
3

Les boucles

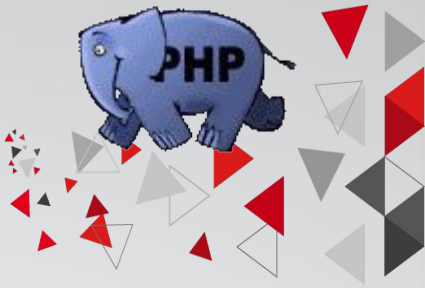
Boucle foreach

```php
<?php
        $retour='<br/>';
    ?>
<!DOCTYPE html>
<html>
<head>
    <title>Document PHP</title>
</head>
<body>
    <?php
    $i=array(1,2,3,4);
    foreach($i as $valeur)
    {
        echo $valeur.$retour;
    }
    ?>
</body>
</html>
```

Résultat:

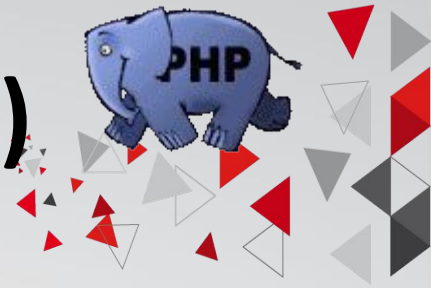
1
2
3
4

Déclaration de fonction




```
Function nom_fonction(){
  instruction1;
  instruction2; ...
}
Exemple:  function calculSomme($a,$b){
Return $a+$b;
}
```

Les fonctions utiles (1/3)



include()

La fonction **include** inclus et exécute le fichier spécifié en argument dans la page.

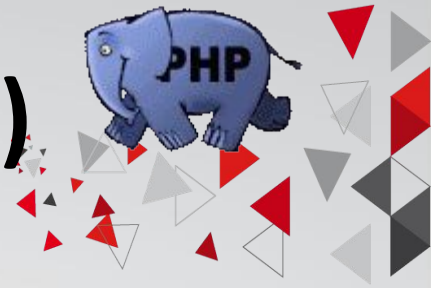
externe.php

```php
<?php
   $a = 35;
   $b = 2;
?>
```

test.php

```php
<?php
   echo "a=$a et b=$b"; // a= b=
   include ("externe.php");
   echo "a=$a et b=$b"; // a=35 b=2
?>
```

Les fonctions utiles (2/3)



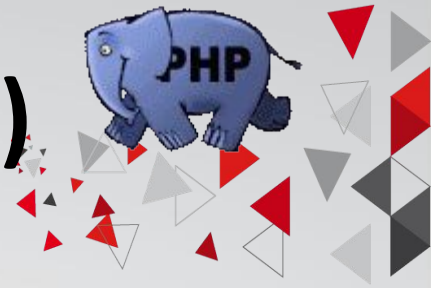
isset()

isset($nom) permet de savoir si la variable $nom existe et initialisé ou pas, isset renvoie TRUE si la variable est définie et FALSE sinon.

Empty()

Empty($nom) détermine si une variable contient une valeur non nulle. Empty retourne la valeur FALSE si la variable nom est affecté ou bien a une valeur différente de 0 et TRUE dans les autres cas.

Les fonctions utiles (3/3)



unset()

unset($nom) détruit une variable

```
//détruire une variable
unset ($x);
//détruire une case d'un tableau
unset ($t[3]);
//détruire plusieurs variables
unset ($x,$y,$z);
```

unset.php

var_dump

Elle affiche le contenu d'une variable ainsi que son type.