

المدرسة الوطنية للعلوم التطبيقية - أكادير  
+٢١٢ ٥٣٥ ٤٠٠ ١١ +٢١٢ ٥٣٥ ٤٠٠ ١١ - ٥٣٥ ٤٠٠ ١١  
ÉCOLE NATIONALE DES SCIENCES APPLIQUÉES - AGADIR



# DÉVELOPPEMENT WEB AVEC PHP

**Sécurité IT Et Confiance Numérique (SITCN)**  
*Ecole nationale des sciences appliquées à Agadir*  
**Semestre 2 - A.U. 2024/2025**  
**Prof: N. BENTAHER**



المدرسة الوطنية للعلوم التطبيقية - أكادير  
+٢١٢ ٥٣٥ ٤٠٠ ١١ +٢١٢ ٥٣٥ ٤٠٠ ١١ - ٥٣٥ ٤٠٠ ١١  
ÉCOLE NATIONALE DES SCIENCES APPLIQUÉES - AGADIR

# Plan du cours

- Introduction à PHP
- Fonctions et tableaux en PHP
- PHP et Formulaire HTML
- PHP et MySQL : Gestion de données avec PHP
- Programmation Orientée Objet (POO) en PHP
- Gestion des erreurs et débogage
- Frameworks PHP
- Sécurité en PHP



# Introduction à PHP



# What is PHP?

PHP (Hypertext Preprocessor) est un langage de script côté serveur principalement utilisé pour le développement web.

- **Caractéristiques principales :**

- ✓ Exécution côté serveur : S'exécute sur le serveur, générant un contenu dynamique avant de l'envoyer au navigateur.
- ✓ Incorporé dans le HTML : PHP peut être mélangé à HTML pour créer des pages web dynamiques.
- ✓ Interagit avec les bases de données : Fonctionne avec MySQL, PostgreSQL, etc.
- ✓ Supporte la programmation orientée objet (POO).
- ✓ Multiplateforme : Fonctionne sous Windows, macOS et Linux.

# Configuration de l'environnement de développement PHP

- **Conditions d'installation**

Pour exécuter PHP, vous avez besoin d'un environnement serveur tel que :

- ✓ XAMPP (Windows, macOS, Linux)
- ✓ MAMP (macOS, Windows)
- ✓ Laragon (Windows)

# Écrire son premier script PHP

- **Un fichier PHP :**
  - A une extension .php (ex: mon\_script.php).
  - Peut contenir du HTML et du PHP.
  - Doit être exécuté sur un serveur local (comme XAMPP, MAMP, ou WAMP).
- Lorsque vous utilisez XAMPP, tous vos fichiers PHP doivent être placés dans le dossier htdocs.
  - Chemin par défaut du dossier htdocs selon le système d'exploitation :
    - Windows : C:\xampp\htdocs\
    - macOS : /Applications/XAMPP/htdocs/
    - Linux : /opt/lampp/htdocs/

## Exécuter le script PHP dans le navigateur

- Démarrer XAMPP et activer Apache
- Ouvrir un navigateur et taper l'URL suivante : [http://localhost/mon\\_script.php](http://localhost/mon_script.php)



# Écrire son premier script PHP

```
<?php  
    echo "Bonjour, ceci est mon premier script PHP !";  
?>
```

# Syntaxe de Base en PHP

## 1. Balises PHP

Le code PHP est encadré par les balises `<?php ... ?>`.

## 2. Commentaires en PHP

- Commentaires sur une ligne : `//` ou `#`
- Commentaires sur plusieurs lignes : `/* ... */`

## 3. Structure de Base d'un Script PHP

Un script PHP commence par `<?php` et se termine par `?>`.

## 4. Règles de Syntaxe

- Les instructions PHP se terminent par un point-virgule (;).
- Les noms de variables sont sensibles à la casse (`$nom`  $\neq$  `$Nom`).
- Les chaînes de caractères peuvent être entourées de guillemets simples (') ou doubles (").



# Variables et Types de Données en PHP

## Définition d'une Variable

- ❖ En PHP une variable est un espace mémoire qui stocke une valeur.
- ❖ En PHP, une variable commence toujours par \$ (ex: \$nom).
- ❖ PHP est un langage faiblement typé : il ne nécessite pas de déclaration explicite du type.

## Règles de Nomination des Variables

- ❖ Doit commencer par \$ suivi d'une lettre ou \_.
- ❖ Ne peut contenir que des lettres, chiffres et \_ (pas d'espaces ou de caractères spéciaux).
- ❖ Sensible à la casse (\$Nom et \$nom sont différentes).

# Variables et Types de Données en PHP

## Types de Données en PHP

### ➤ Types scalaires :

- ✓ string : Chaîne de caractères (\$nom = "PHP";)
- ✓ int : Nombre entier (\$age = 25;)
- ✓ float : Nombre à virgule (\$prix = 9.99;)
- ✓ bool : Booléen (\$actif = true;)

### ➤ Types composés :

- ✓ array : Tableau (\$nombres = array(1, 2, 3);)
- ✓ object : Objet (class Voiture { })

### ➤ Types spéciaux :

- ✓ NULL : Absence de valeur (\$val = null;)
- ✓ resource : Référence externe (ex: base de données, fichier)

# Variables et Types de Données en PHP

## ➤ Affectation et Modification des Variables

- Affectation simple : `$nom = "Alice";`
- Modification : `$nom = "Bob";`
- Concaténation : `$message = "Bonjour " . $nom;`

## ➤ Conversion de Types (Casting)

- `(int) "123" => 123`
- `(float) "12.5" => 12.5`
- `(string) 100 => "100"`
- `(bool) 0 => false`

# Variables et Types de Données en PHP

## ➤ Vérification et Manipulation des Types

- `gettype($var)` : Retourne le type d'une variable.
- `is_int($var)`, `is_string($var)`, `is_array($var)`, etc.
- `var_dump($var)` : Affiche les détails d'une variable.

## ➤ Portée des Variables

- Locale : Défini à l'intérieur d'une fonction.
- Globale : Accessible partout avec `global` ou `$GLOBALS`.
- Statique : Conserve sa valeur entre les appels (`static $compteur = 0;`).

# Variables et Types de Données en PHP

```
<?php
// Définition de différentes variables
$var1 = 42;
$var2 = "Bonjour";
$var3 = [1, 2, 3];
$var4 = 3.14;
$var5 = true;
```

- ◆ `is_int()` :

var1: 1

- ◆ `is_string()` :

var2: 1

- ◆ `is_array()` :

var3: 1

- ◆ `gettype()` :

var1 : integer

var2 : string

var3 : array

var4 : double

var5 : boolean

- ◆ `var_dump()` :

var1 : int(42)

var2 : string(7) "Bonjour"

var3 : array(3) { [0]=> int(1) [1]=> int(2) [2]=> int(3) }

# Les Constantes en PHP

Une constante est une variable dont la valeur ne peut pas être modifiée après déclaration.

Déclaration avec *define()* ou *const*

Exemple:

```
define("PI", 3.1416);  
const TVA = 0.2;  
echo PI; // Affiche 3.1416
```



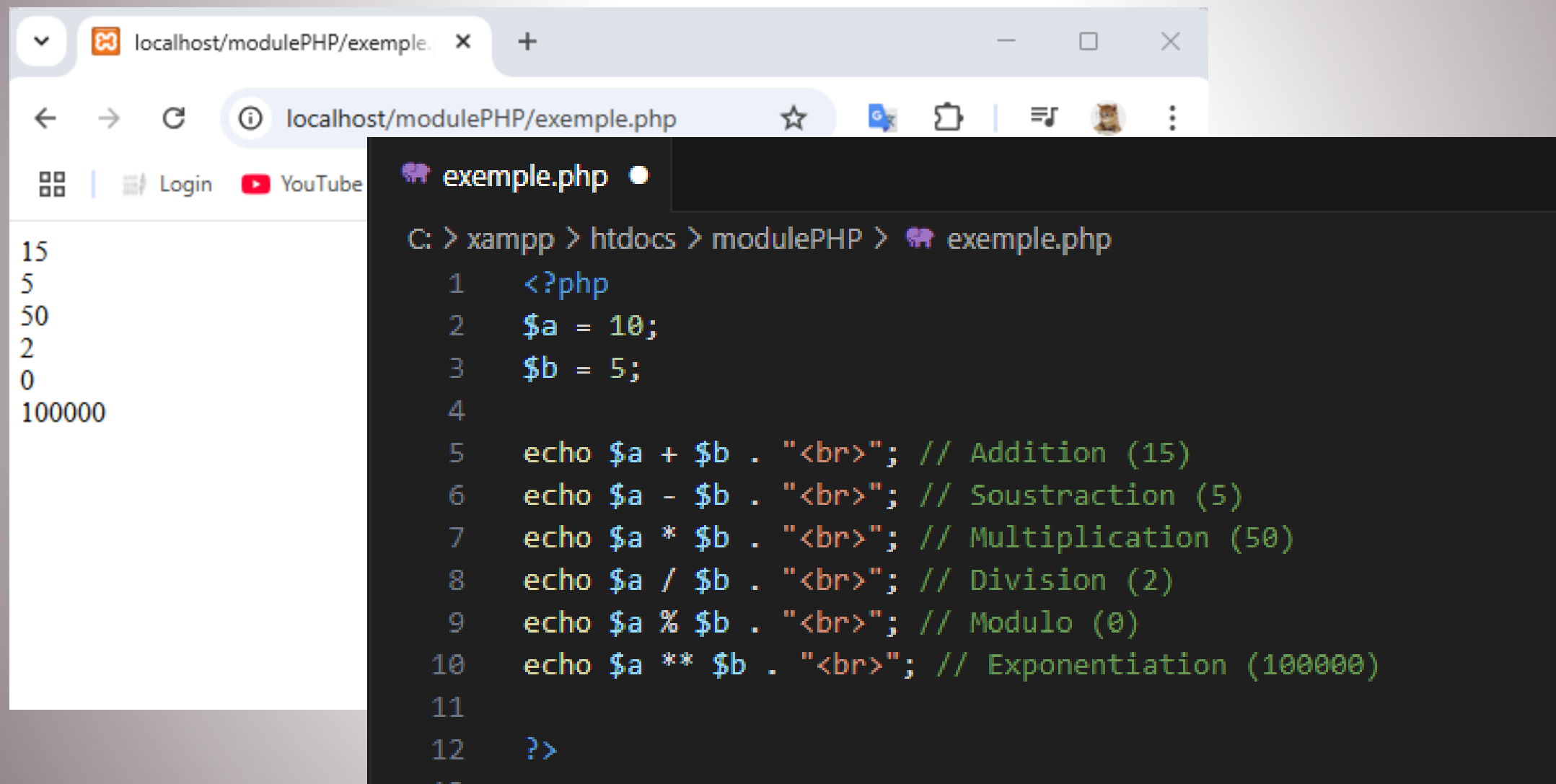
# Opérateurs en PHP

- Les opérateurs sont utilisés pour effectuer des opérations sur des variables et des valeurs.
- PHP supporte plusieurs types d'opérateurs : arithmétiques, de comparaison, logiques, etc.

## Opérateurs Arithmétiques

- Utilisés pour effectuer des calculs mathématiques.

# Opérateurs en PHP



The image shows a web browser window on the left and a code editor on the right. The browser displays the output of a PHP script: 15, 5, 50, 2, 0, and 100000, each on a new line. The code editor shows the source code of the script, which uses PHP operators to calculate these values. The code is as follows:

```
C: > xampp > htdocs > modulePHP > exemple.php
1  <?php
2  $a = 10;
3  $b = 5;
4
5  echo $a + $b . "<br>"; // Addition (15)
6  echo $a - $b . "<br>"; // Soustraction (5)
7  echo $a * $b . "<br>"; // Multiplication (50)
8  echo $a / $b . "<br>"; // Division (2)
9  echo $a % $b . "<br>"; // Modulo (0)
10 echo $a ** $b . "<br>"; // Exponentiation (100000)
11
12 ?>
```

# Opérateurs en PHP

## Opérateurs de Comparaison

- Utilisés pour comparer deux valeurs.
- Retournent true (vrai) ou false (faux).

## Opérateurs Logiques

- Utilisés pour combiner des conditions.
- Retournent true ou false.

```
$a = true;  
$b = false;  
  
echo $a && $b; // ET logique (false)  
echo $a || $b; // OU logique (true)  
echo !$a;      // NON logique (false)
```

```
echo $x == $y; // Égal à (false)  
echo $x != $y; // Différent de (true)  
echo $x > $y;  // Supérieur à (true)  
echo $x < $y;  // Inférieur à (false)  
echo $x >= $y; // Supérieur ou égal à (true)  
echo $x <= $y; // Inférieur ou égal à (false)  
echo $x === $y; // Identique (même type et valeur, false)  
echo $x !== $y; // Non identique (true)
```

# Les Structures Conditionnelles en PHP

Les instructions conditionnelles permettent d'exécuter du code en fonction d'une condition.


## Types de conditions en PHP :

✓ if...else : Permet d'exécuter un bloc de code si une condition est vraie.

```
C: > xampp > htdocs > boucles.php
1  <?php
2  $age = 18;
3
4  ✓ if ($age >= 18) {
5      |     echo "<p>Vous êtes majeur.</p>";
6  ✓ } else {
7      |     echo "<p>Vous êtes mineur.</p>";
8      |
9      ?>
10
```

# Les Structures Conditionnelles en PHP

- ✓ if...elseif...else : Permet de tester plusieurs conditions successives.

```
C: > xampp > htdocs >  boucles.php
1  <?php
2  $note = 15;
3
4  ✓ if ($note >= 16) {
5      |     echo "<p>Très bien !</p>";
6  ✓ } elseif ($note >= 12) {
7      |     echo "<p>Assez bien.</p>";
8  ✓ } elseif ($note >= 10) {
9      |     echo "<p>Passable.</p>";
10 ✓ } else {
11     |     echo "<p>Échec.</p>";
12     |
13     ?>
14
```

# Les Structures Conditionnelles en PHP

- ✓ Switch: Alternative à if...elseif...else, utile pour tester une seule variable contre plusieurs valeurs.

```
C: > xampp > htdocs > 🐘 boucles.php
1  <?php
2  $jour = "mardi";
3
4  switch ($jour) {
5      case "lundi":
6          echo "<p>📅 C'est le début de la semaine.</p>";
7          break;
8      case "mercredi":
9          echo "<p>📅 C'est le milieu de la semaine.</p>";
10         break;
11     case "vendredi":
12         echo "<p>🎉 Bientôt le week-end !</p>";
13         break;
14     default:
15         echo "<p>📅 Un jour comme un autre.</p>";
16     }
17     ?>
```



# Les Structures Conditionnelles en PHP

## Bonnes Pratiques

- ✓ Toujours utiliser **break** dans un **switch** pour éviter l'exécution des cas suivants.
- ✓ **Préférer if...else** pour des comparaisons complexes.
- ✓ Utiliser switch lorsque la variable peut prendre plusieurs valeurs fixes.

# Les Boucles en PHP

Les boucles permettent d'exécuter un bloc de code plusieurs fois selon une condition.

## **Types de boucles en PHP :**

- for
- while
- do...while
- foreach

# Les Boucles en PHP

- For : – Répète un bloc de code un nombre défini de fois, Utilisée lorsque le nombre d'itérations est connu à l'avance.

```
for ($i = 1; $i <= 5; $i++) {  
    echo "Itération n°" . $i . "<br>";  
}
```

# Les Boucles en PHP

- While: Répète un bloc de code tant qu'une condition est vraie, Utilisée lorsque le nombre d'itérations est inconnu et dépend d'une condition.

```
$x = 1;

while ($x <= 5) {
    echo "Valeur de x : " . $x . "<br>";
    $x++;
}
```

# Les Boucles en PHP

- do...while : Exécute d'abord le code puis vérifie la condition, le code est exécuté au moins une fois.

```
do {  
    echo "Valeur de y : " . $y . "<br>";  
    $y++;  
} while ($y <= 5);
```

# Les Boucles en PHP

- Foreach: Parcourt les tableaux et objets

```
$fruits = ["Pomme", "Banane", "Orange"];  
  
foreach ($fruits as $fruit) {  
    echo "Fruit : " . $fruit . "<br>";  
}
```



# Fonctions et tableaux en PHP



# Qu'est-ce qu'une Fonction ?

Une fonction est un bloc de code réutilisable qui effectue une tâche spécifique. Elle permet d'éviter la répétition du code, d'améliorer la lisibilité et d'optimiser la maintenance d'un programme. En PHP, une fonction est définie avec le mot-clé *function*, suivi de son nom et d'éventuels paramètres.

Les fonctions peuvent être utilisées pour structurer le code en différentes parties logiques, facilitant ainsi la compréhension et la modification du programme. Elles peuvent recevoir des arguments et retourner des résultats, ce qui les rend très flexibles.

# Qu'est-ce qu'une Fonction ?

## **Il existe plusieurs types de fonctions en PHP :**

- Les fonctions simples, qui exécutent une action sans paramètres.
- Les fonctions avec paramètres, qui acceptent des valeurs pour traiter des données dynamiquement.
- Les fonctions qui retournent une valeur, qui permettent de récupérer un résultat et de l'utiliser ailleurs dans le code.
- Les fonctions anonymes (closures), qui peuvent être stockées dans des variables ou passées comme arguments à d'autres fonctions.
- Les fonctions fléchées, introduites en PHP 7.4, qui sont une version simplifiée des fonctions anonymes.

# Qu'est-ce qu'une Fonction ?

exemple.php X

C: > xampp > htdocs > modulePHP > exemple.php

```
1  <?php
2  // Definition d'une fonction simple
3  function direBonjour() {
4      echo "Bonjour !";
5  }
6
7  // Appel de la fonction
8  direBonjour();
9
10 // Fonction avec parametre
11 function saluer($nom) {
12     echo "Bonjour, " . $nom . " !";
13 }
14
15 saluer("Alice");
16
```

# Qu'est-ce qu'une Fonction ?

```
// Fonction qui retourne une valeur
function additionner($a, $b) {
    return $a + $b;
}

$resultat = additionner(5, 3);
echo $resultat;

// Fonction avec valeur par défaut
function bienvenue($nom = "Invité") {
    echo "Bienvenue, " . $nom . " !";
}

bienvenue();
bienvenue("Sophie");
```

# Qu'est-ce qu'une Fonction ?

```
// Fonction avec nombre variable d'arguments
function somme(...$nombres) {
    return array_sum($nombres);
}

echo somme(1, 2, 3, 4);

// Fonction anonyme
$direBonjour = function($nom) {
    return "Bonjour, " . $nom . " !";
};

echo $direBonjour("Lucas");

// Fonction flechee (PHP 7.4+)
$square = fn($x) => $x * $x;
echo $square(4);
```



# Portée des Variables en PHP (Global, Locale, Statique)

La portée d'une variable détermine où elle peut être utilisée dans le script. En PHP, il existe trois types principaux de portée :

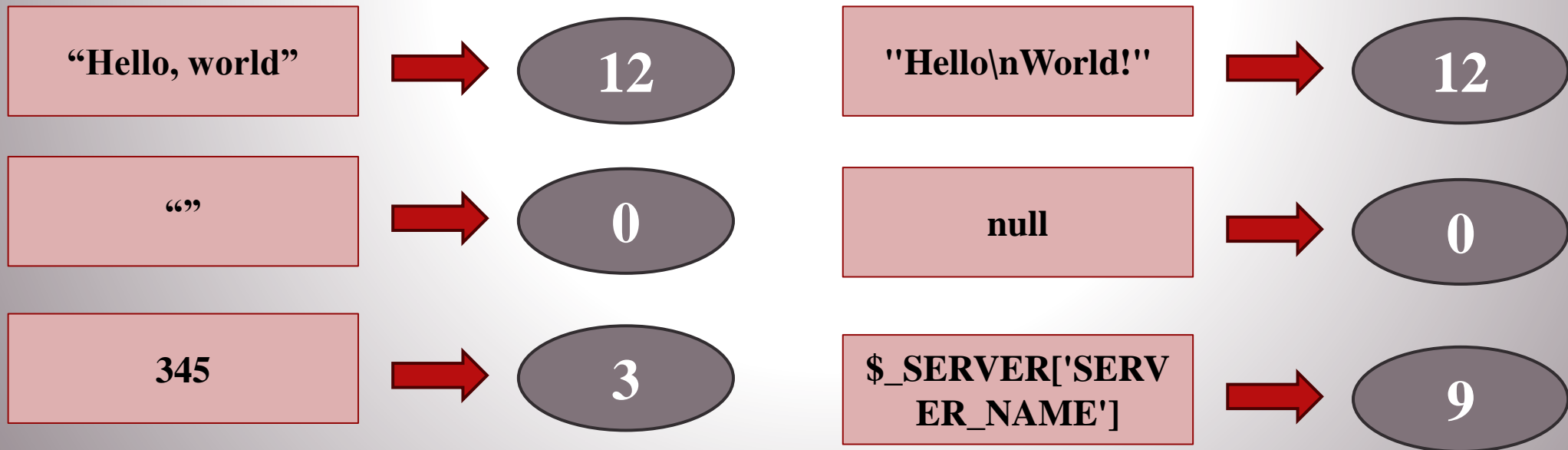
- **Portée locale** : Une variable déclarée à l'intérieur d'une fonction n'est accessible qu'à l'intérieur de cette fonction.
- **Portée globale** : Une variable déclarée en dehors d'une fonction ne peut pas être directement utilisée à l'intérieur de cette fonction, sauf si elle est déclarée avec le mot-clé global.
- **Portée statique** : Une variable déclarée avec static conserve sa valeur entre les appels successifs de la fonction.

# Les Fonctions Intégrées en PHP (String, Math, Date)

## 1. Fonctions de Manipulation des Chaînes (String)

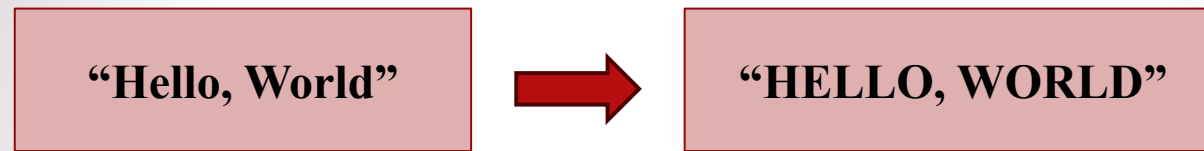
PHP offre plusieurs fonctions intégrées pour manipuler les chaînes de caractères :

✓ `strlen($str)`: Retourne la longueur d'une chaîne.

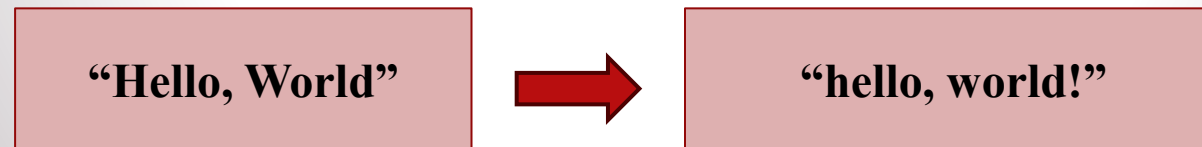


# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `strtoupper($str)`: Convertit une chaîne en majuscules.

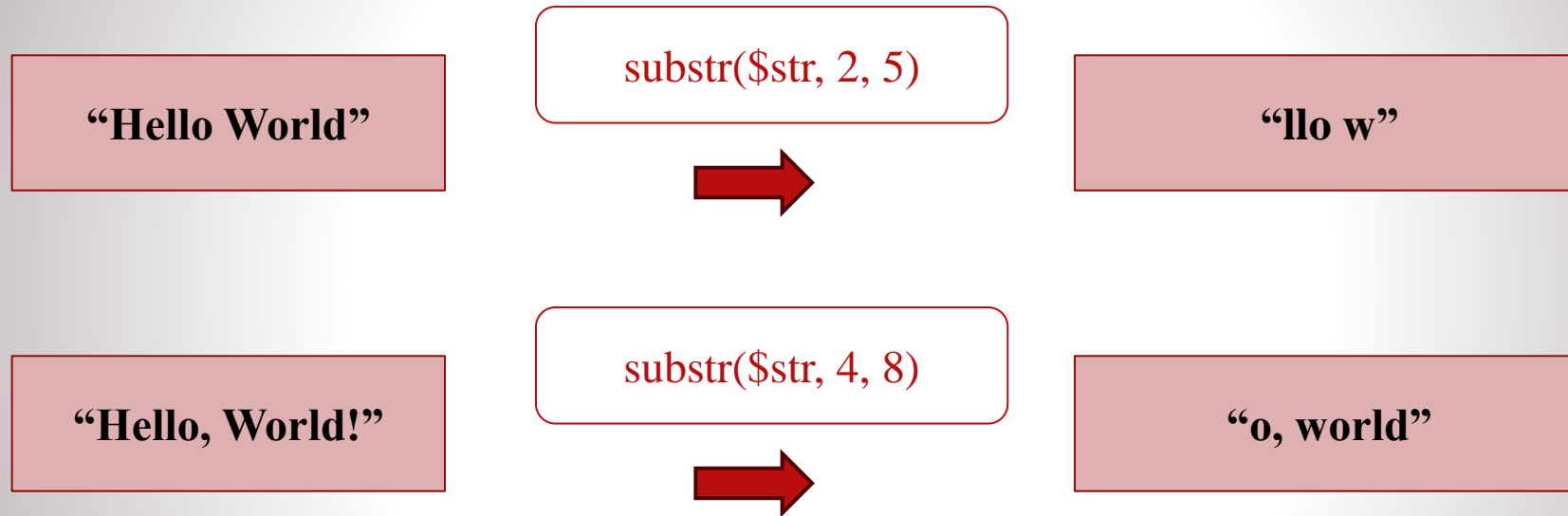


- ✓ `strtolower($str)`: Convertit une chaîne en minuscules.



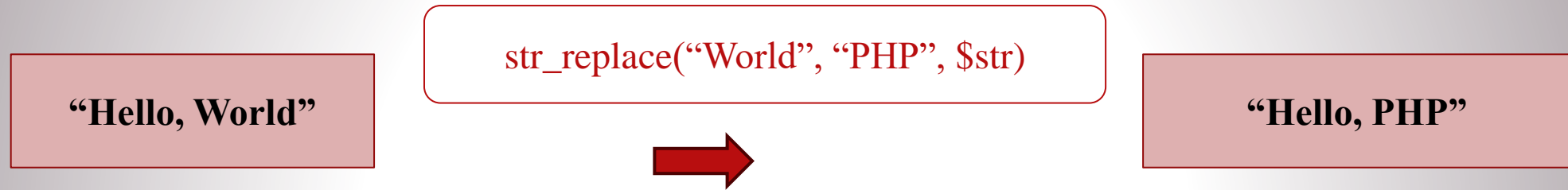
# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `substr($str, $start, $length)`: Extrait une partie de la chaîne.

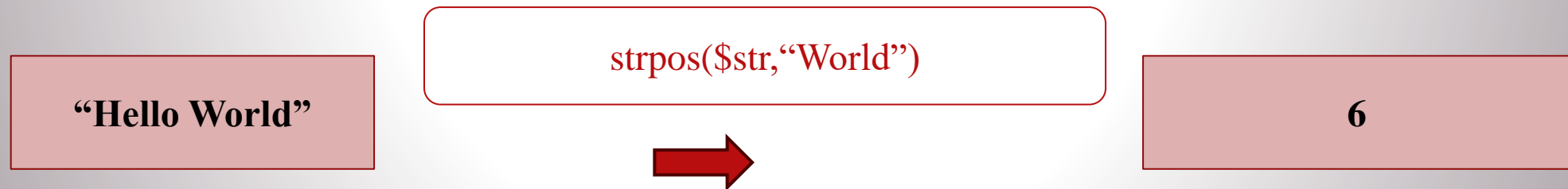


# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `str_replace($search, $replace, $str)`: Remplace un mot dans une chaîne.



- ✓ `strpos($str, $word)`: Trouve la position d'un mot dans une chaîne.

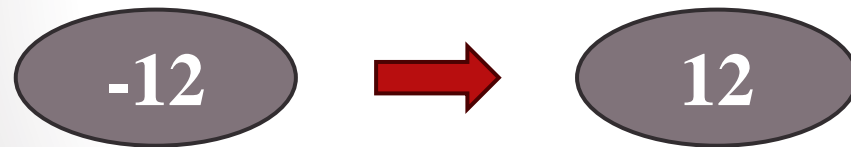


# Les Fonctions Intégrées en PHP (String, Math, Date)

## 2. Fonctions Mathématiques (Math)

PHP propose plusieurs fonctions pour les calculs mathématiques :

- ✓ `abs($num)`: Retourne la valeur absolue d'un nombre.



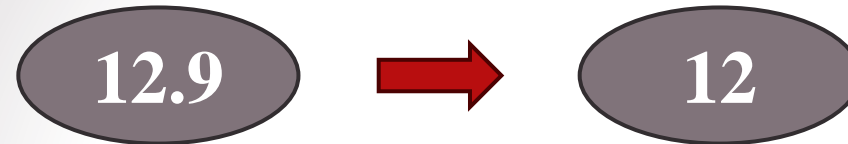
- ✓ `round($num)`: Arrondit un nombre.



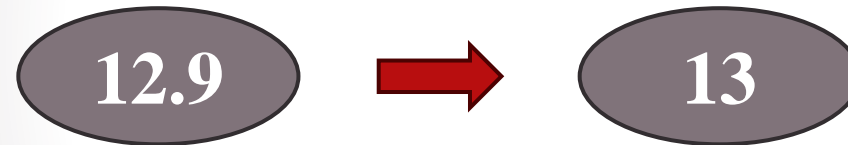


# Les Fonctions Intégrées en PHP (String, Math, Date)

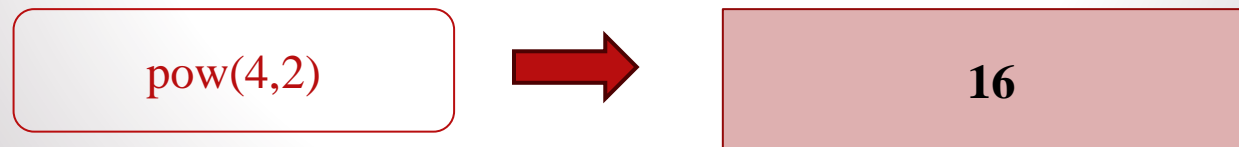
- ✓ `floor($num)`: Arrondit vers le bas.



- ✓ `ceil($num)`: Arrondit vers le haut.



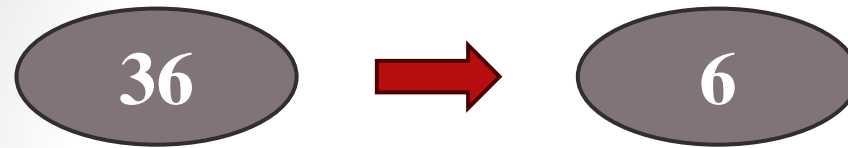
- ✓ `pow($base, $exp)`: Calcule la puissance d'un nombre.



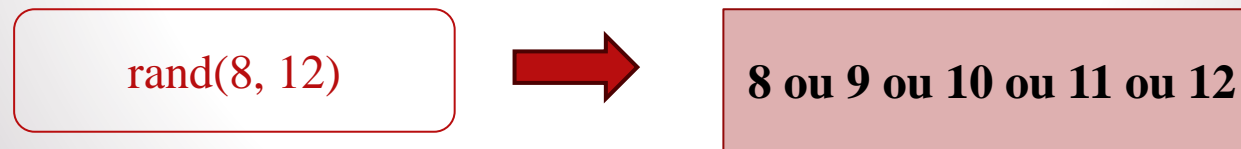


# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `sqrt($num)`: Retourne la racine carrée d'un nombre.



- ✓ `rand($min, $max)`: Génère un nombre aléatoire entre \$min et \$max.



# Les Fonctions Intégrées en PHP (String, Math, Date)

## 3. Fonctions de Gestion des Dates (Date)

PHP fournit plusieurs fonctions pour manipuler les dates et heures :

- ✓ `date($format)`: Retourne la date sous un format spécifique.
  - ✓ `date("Y-m-d H:i:s");`
- ✓ Convertir un Timestamp en Date Lisible

```
date("Y-m-d H:i:s", 1741459200)
```



```
2025-03-09 12:00:00
```

# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `time()`: Retourne le timestamp actuel.
- ✓ Un **timestamp** est un entier représentant le nombre de **secondes** écoulées depuis le **1er janvier 1970 à 00:00:00 UTC** (appelé aussi **l'époque Unix**). Il est souvent utilisé pour gérer les dates et heures en programmation.



# Les Fonctions Intégrées en PHP (String, Math, Date)

- ✓ `strtotime($string)`: Convertit une date en timestamp.

`strtotime("2025-03-09 12:00:00")`



`1741459200`

- ✓ C'est le timestamp pour le 9 mars 2025 à 12:00 UTC.

- ✓ `mktime($h, $m, $s, $mois, $jour, $année)`: Retourne un timestamp à partir d'une date donnée.

`mktime(12, 0, 0, 3, 9, 2025)`



`1741502400`

# Introduction aux Tableaux (Arrays) en PHP

Un tableau (array) est une structure de données permettant de stocker plusieurs valeurs sous une seule variable. Chaque valeur est associée à une clé unique, qui peut être un entier (indexé) ou une chaîne de caractères (associatif).

Les tableaux sont utilisés pour organiser et manipuler des ensembles de données de manière efficace.

# Introduction aux Tableaux (Arrays) en PHP

## Types de Tableaux en PHP

- Tableaux Indexés : Les clés sont des entiers commençant par 0.

```
$couleurs = ["Rouge", "Vert", "Bleu"];  
echo $couleurs[1];
```

**Resultat**

Vert

```
$animaux = ["Chat", "Chien", "Lapin"];  
echo $animaux[2];
```

**Resultat**

Lapin

# Introduction aux Tableaux (Arrays) en PHP

- Tableaux Associatifs : Les clés sont des chaînes de caractères.

```
$personne = ["nom" => "Alice", "age" => 25, "ville" => "Paris"];  
  
echo $personne["ville"];
```

**Resultat**

Paris

```
$couleurs = ["rouge" => "#FF0000", "vert" => "#00FF00", "bleu" => "#0000FF"];  
  
echo $couleurs["vert"];
```

**Resultat**

#00FF00



# Introduction aux Tableaux (Arrays) en PHP

- Tableaux Multidimensionnels : Des tableaux imbriqués dans d'autres tableaux.

```
$films = [  
    "Action" => ["Mad Max", "John Wick"],  
    "Comédie" => ["Mr. Bean", "Dumb and Dumber"]  
];  
echo $films["Comédie"][0];
```



**Resultat**

Mr. Bean

# Introduction aux Tableaux (Arrays) en PHP

## Parcourir un Tableau

- ◆ Avec foreach (le plus utilisé)

```
$fruits = ["Pomme", "Banane", "Orange"];  
foreach ($fruits as $fruit) { echo $fruit . "<br>"; }
```

# Introduction aux Tableaux (Arrays) en PHP

## Parcourir un Tableau

```
$fruits = ["Pomme", "Banane", "Orange"];  
foreach ($fruits as $fruit) {  
    echo $fruit . " ";  
}
```

1- Pomme Banane Orange



2- Pomme, Banane, Orange



3- 0 1 2



4- Erreur



```
$personne = ["Nom" => "Alice", "Âge" =>  
25];
```

```
foreach ($personne as ???) {  
    echo "$cle: $valeur <br>";  
}
```

1- foreach (\$personne as \$valeur)



2- foreach (\$personne as \$cle =>  
\$valeur)



3- foreach (\$valeur as \$cle =>  
\$personne)



4- foreach (\$personne)



# Introduction aux Tableaux (Arrays) en PHP

## Parcourir un Tableau

```
$fruits = ["Pomme", "Banane", "Orange"];  
foreach ($fruits as ???) {  
    echo "$index: $fruit <br>";  
}
```

1- foreach (\$fruits as \$index => \$fruit)



2- foreach (\$fruits as \$fruit => \$index)



3- foreach (\$index as \$fruits => \$fruit)



4- Aucune de ces réponses



```
$chiffres = [10, 20, 30];  
foreach ($chiffres as $chiffre) {  
    $chiffre *= 2;  
}  
print_r($chiffres);
```

1- [20, 40, 60]



2- [10, 20, 30]



3- [10, 10, 10]



4- Erreur



# Introduction aux Tableaux (Arrays) en PHP

## Parcourir un Tableau

Comment modifier les valeurs du tableau directement dans foreach ?

1- `foreach ($tableau as &$valeur) { $valeur *= 2; }`



2- `foreach ($tableau as $valeur) { $valeur *= 2; }`



3- `foreach (&$tableau as $valeur) { $valeur *= 2; }`



4- Aucune de ces réponses



# Les Fonctions Utiles pour les Tableaux en PHP

## 1. array\_map(): Appliquer une Fonction à Chaque Élément

```
$couleurs = ["rouge", "bleu", "vert"];  
array_map(function($couleur) {  
    return strtoupper($couleur);  
}, $couleurs);  
  
print_r($couleurs);
```

```
$nombres = [5, 10, 15];  
$double = array_map(fn($n) => $n * 2,  
$nombres);  
echo $double[1];
```

1- Array ( [0] => ROUGE [1] => BLEU [2] => VERT )



2- Array ( [0] => rouge [1] => bleu [2] => vert )



3- Erreur



4- Aucune de ces réponses



1-10



2- 20



3- 30



4- Erreur



# Introduction aux Tableaux (Arrays) en PHP

Quelle est la différence entre foreach et array\_map() ?

1-foreach modifie directement le tableau, alors que array\_map() crée un nouveau tableau.



2-array\_map() ne fonctionne qu'avec des tableaux multidimensionnels.



3-Il n'y a aucune différence, ils font exactement la même chose.



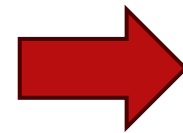


# Les Fonctions Utiles pour les Tableaux en PHP

## 2. array\_filter() – Filtrer un Tableau

array\_filter() permet de filtrer un tableau en supprimant les éléments qui ne respectent pas une condition.

```
$mots = ["chat", "chien", "rat",  
"éléphant"];  
$motsLongs = array_filter($mots, fn($mot)  
=> strlen($mot) > 3);  
  
print_r($motsLongs);
```



```
Array ( [0] => chat [1] =>  
chien [3] => éléphant )
```

# Les Fonctions Utiles pour les Tableaux en PHP

## 2. array\_filter()

```
$notes = [9, 15, 8, 20, 13];  
$admis = array_filter($notes, fn($n) => $n >= 10);  
print_r($admis);
```

1-Array ( [0] => 15 [1] => 20 [2] => 13 )



2- Array ( [1] => 15 [3] => 20 [4] => 13 )



3- Erreur



# Les Fonctions Utiles pour les Tableaux en PHP

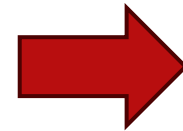
## Explication :

- `array_filter()` supprime les valeurs qui ne respectent pas la condition ( $\geq 10$ ).
- Les nombres 9 et 8 sont supprimés.
- Les clés originales sont conservées (1, 3, 4).

# Les Fonctions Utiles pour les Tableaux en PHP

**Pour réindexer les clés, ajoute `array_values($admis);`**

```
$admis = array_values($admis);  
print_r($admis);
```



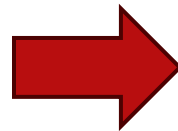
```
Array ( [0] => 15 [1] => 20 [2]  
=> 13 )
```

# Les Fonctions Utiles pour les Tableaux en PHP

## 3. array\_merge() – Fusionner des Tableaux

array\_merge() combine plusieurs tableaux en un seul.

```
$a = [1, 2, 3];  
$b = [4, 5, 6];  
$result = array_merge($a, $b);  
  
print_r($result);
```

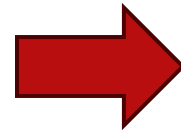


```
Array ( [0] => 1 [1] => 2 [2] =>  
3 [3] => 4 [4] => 5 [5] => 6 )
```

Les valeurs des tableaux sont fusionnées et les clés numériques sont réindexées.

# Les Fonctions Utiles pour les Tableaux en PHP

```
$a = ["nom" => "Alice", "age" => 25];  
$b = ["age" => 30, "ville" => "Paris"];  
$result = array_merge($a, $b);  
  
print_r($result);
```



```
Array ( [nom] => Alice [age] =>  
30 [ville] => Paris )
```

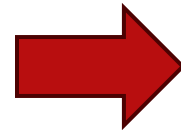
"nom" reste inchangé.

"age" remplace 25 par 30 (valeur du 2<sup>e</sup> tableau).

"ville" est ajoutée.

# Les Fonctions Utiles pour les Tableaux en PHP

```
$a = [1, "nom" => "Alice"];  
$b = ["age" => 25, 2 => "Paris"];  
$result = array_merge($a, $b);  
  
print_r($result);
```



```
Array ( [0] => 1 [nom] => Alice  
[age] => 25 [1] => Paris )
```

Les clés associatives sont fusionnées normalement.

Les clés numériques sont réindexées (1 devient 0, 2 devient 1).



# Les Fonctions Utiles pour les Tableaux en PHP

```
$x = ["a", "b"];  
$y = [1 => "c", 2 => "d"];  
$result = array_merge($x, $y);  
print_r($result);
```

1- Array ( [0] => a [1] => b [2] => c [3] => d )



2- Array ( [0] => a [1] => b [1] => c [2] => d )



3- Erreur



`$x = ["a", "b"]` → Il s'agit d'un tableau indexé

`$y = [1 => "c", 2 => "d"]`; Tableau avec indices personnalisés (1 et 2 au lieu de 0 et 1)

`array_merge()` fusionne les tableaux et réindexe toutes les clés numériques en commençant par 0.

# Tri des Tableaux en PHP

1. `sort()` – Trier un Tableau en Ordre Croissant
  - ◆ La fonction `sort()` trie un tableau indexé en ordre croissant (du plus petit au plus grand).
2. `rsort()` – Trier un Tableau en Ordre Décroissant
3. `asort()` – Trier un Tableau Associatif par Valeurs (Ordre Croissant)
4. `ksort()` – Trier un Tableau Associatif par Clés (Ordre Croissant)

# Les Tableaux Superglobaux en PHP

Les tableaux superglobaux sont des variables intégrées disponibles dans tous les contextes d'un script PHP, peu importe où vous vous trouvez (fonction, classe, fichier, etc.). Ils sont utilisés pour stocker et récupérer des informations entre les pages ou lors des interactions avec un utilisateur.

## 1. `$_GET` – Récupérer les Données dans l'URL

# Les Tableaux Superglobaux en PHP

```
//index.php?nom=Bob&age=30  
  
$nom = $_GET['nom'];  
$age = $_GET['age'];  
echo "Nom : $nom, Age : $age";
```

Nom : Alice, Age : 25



Nom : Bob, Age : 30



Erreur



```
//index.php?age=40  
$nom = $_GET['nom'];  
$age = $_GET['age'];  
  
echo "Nom : $nom, Age : $age";
```

Nom : , Age : 40



Erreur Undefined index 'nom'



Nom : NULL, Age : 40



Quelle est la meilleure façon d'éviter une erreur si nom ou age est absent ?

`isset($_GET['nom']) ? $_GET['nom'] : 'Inconnu'`



`$_GET['nom']`



`empty($_GET['nom'])`



`isset($_GET['nom'])` vérifie si la clé "nom" existe dans `$_GET`.

Si elle existe, elle est utilisée. Sinon, on affiche "Inconnu".

# Les Tableaux Superglobaux en PHP

## 2. **\$\_POST** – Récupérer les Données dans un Formulaire

**\$\_POST** est utilisé pour récupérer des données envoyées par la méthode POST (généralement utilisée dans les formulaires HTML, les fichiers, etc.).

```
<form method="POST" action="index.php">
  <input type="text" name="nom"
value="Alice">
  <input type="submit" value="Envoyer">
</form>
```

```
//code php
```

```
$nom = $_POST['nom'];
echo "Nom : $nom";
```

Nom : Alice



Nom :



Undefined index: nom



Que se passe-t-il si l'utilisateur soumet le formulaire sans entrer de valeur ?

Nom :



Undefined index: nom



Erreur fatale





Comment éviter l'erreur si nom n'est pas défini ?

```
isset($_POST['nom']) ? $_POST['nom'] : 'Inconnu'
```



```
$_POST['nom']
```



```
empty($_POST['nom'])
```



`isset($_POST['nom'])` vérifie si nom est défini.

Si nom existe, on affiche sa valeur.

Sinon, on affiche "Inconnu".

Quelle est la principale différence entre `$_GET` et `$_POST` ?

`$_GET` est plus sécurisé que `$_POST`



`$_POST` ne stocke pas les données dans l'URL, contrairement à `$_GET`



Il n'y a aucune différence



`$_GET` envoie les données via l'URL (index.php?nom=Alice).

`$_POST` envoie les données dans le corps de la requête HTTP (non visible dans l'URL)

# Les Tableaux Superglobaux en PHP

## 3. **\$\_SESSION** – Stocker des Données pour une Session


**\$\_SESSION** permet de stocker des informations qui peuvent être utilisées tout au long de la session d'un utilisateur. Cela permet de maintenir un état entre différentes pages.

```
// Démarrer la session  
session_start();
```


```
$_SESSION['nom'] = "Alice"; // Stockage d'une donnée
```

```
// Utilisation de la donnée  
echo "Nom en session : " . $_SESSION['nom'];
```


**Quel est le but principal de \$\_SESSION en PHP ?**

- a) Stocker temporairement des données uniquement sur une page
- b) Stocker des informations accessibles à travers plusieurs pages 
- c) Créer une base de données locale
- d) Stocker les préférences de l'utilisateur de manière permanente


**Quelle fonction doit être appelée avant d'utiliser \$\_SESSION ?**

- a) session\_destroy();
- b) session\_start(); 
- c) session\_unset();
- d) session\_register();

**Quelle ligne de code permet d'enregistrer un prénom dans une session PHP ?**

- a) \$\_SESSION["prenom"] = "Alice"; 
- b) session\_start("Alice");
- c) \$\_SESSION->prenom = "Alice";
- d) session\_add("prenom", "Alice");

**Comment supprimer complètement toutes les variables de session en PHP ?**

- a) session\_reset();b)
- b) unset(\$\_SESSION);
- c) session\_destroy();
- d) \$\_SESSION = array(); session\_destroy(); 

# Les Tableaux Superglobaux en PHP


## 4. **\$\_COOKIE** – Stocker des Données dans des Cookies

**\$\_COOKIE** permet de stocker des informations dans des cookies qui persisteront au-delà de la session PHP, même après la fermeture du navigateur.

```
// Créer un cookie (expiration dans 1 jour)
setcookie('user', 'Alice', time() + 86400);


// Récupérer la valeur du cookie
if (isset($_COOKIE['user'])) {
    echo "Utilisateur : " . $_COOKIE['user'];
}
```

**Quel est le principal objectif des cookies en PHP ?**

- A) Stocker des données temporaires uniquement pour la session active
- B) Sauvegarder des informations même après la fermeture du navigateur 
- C) Exécuter du code PHP côté client
- D) Se connecter automatiquement à un serveur

**Contrairement à `$_SESSION`, qui est supprimé après la fermeture du navigateur, les cookies permettent de conserver les données sur le long terme.**

**Quel code permet de créer un cookie en PHP qui dure 30 jours ?**

- A) `session_start(); $_SESSION["user"] = "Alice";`
- B) `setcookie("user", "Alice", time() + (30 * 24 * 60 * 60), "/");` 
- C) `$_COOKIE["user"] = "Alice";`
- D) `$cookie = new Cookie(); $cookie->set("user", "Alice");`

**La fonction `setcookie()` crée un cookie nommé "user" avec la valeur "Alice", une durée de 30 jours et une accessibilité sur tout le site ("/").**



# Les Tableaux Superglobaux en PHP

## Différences entre \$\_GET et \$\_POST

- ❖ \$\_GET est utilisé pour envoyer des données via l'URL. Cela signifie que les informations sont visibles dans la barre d'adresse du navigateur.
- ❖ \$\_POST est utilisé pour envoyer des données via une requête HTTP POST, qui n'affiche pas les informations dans l'URL. C'est plus sécurisé pour des données sensibles.
- ❖ Avant de traiter les données envoyées via \$\_GET ou \$\_POST, il est toujours important de vérifier si elles sont présentes pour éviter des erreurs.
- ❖ Les données envoyées via \$\_GET et \$\_POST peuvent être manipulées par l'utilisateur, il est donc important de les sécuriser avant de les utiliser, notamment contre les attaques telles que **l'injection SQL** ou le **cross-site scripting (XSS)**.



# Exercice

- Écrivez un script PHP qui permet à un utilisateur d'entrer une température de l'eau via un formulaire. Le script doit analyser la température et afficher l'état de l'eau selon les règles suivantes :
  - Température  $\leq 0^{\circ}\text{C}$   $\rightarrow$  L'eau est solide (glace)
  - Entre  $1^{\circ}\text{C}$  et  $99^{\circ}\text{C}$   $\rightarrow$  L'eau est liquide
  - Température  $\geq 100^{\circ}\text{C}$   $\rightarrow$  L'eau est gazeuse (vapeur)

# Exercice pour tester (my\_script.php)

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>État de l'eau</title>
</head>
<body>

  <h2>Déterminez l'état de l'eau en fonction de sa température</h2>
  <form method="POST">
    <label for="temperature">Entrez la température de l'eau (°C) :</label>
    <input type="number" name="temperature" required>
    <button type="submit">Vérifier</button>
  </form>
```

```
<?php
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    // Récupération et conversion de la température en entier  
    $temperature = (int) $_POST["temperature"];  
  
    // Vérification de l'état de l'eau  
    if ($temperature <= 0) {  
        echo "<p>L'eau est solide (glace) à $temperature°C.</p>";  
    } elseif ($temperature >= 100) {  
        echo "<p>L'eau est gazeuse (vapeur) à $temperature°C.</p>";  
    } else {  
        echo "<p>L'eau est liquide à $temperature°C.</p>";  
    }  
}  
?>
```

```
</body>
```

```
</html>
```