

# Php

## PHP: Hypertext Preprocessor



Rachid EDJEKOUANE (edjek@hotmail.fr)



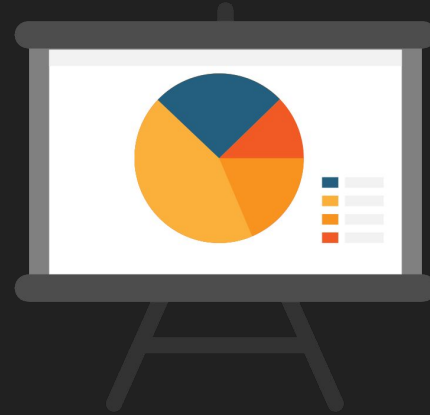
# Prérequis

- Notions d'Algorithmique
- Base de la programmation
- Compréhension des concepts de base de données
- Base HTML / CSS (côté navigateur)
- Ponctualité





# Introduction





# Site dynamique

Un site **dynamique** est un site web dont le contenu et les fonctionnalités sont générés dynamiquement à partir d'une source de données, généralement une base de données.

Contrairement à un site statique, où le contenu est préalablement créé et stocké dans des fichiers HTML individuels, un site dynamique permet de fournir du **contenu personnalisé et interactif** aux utilisateurs.



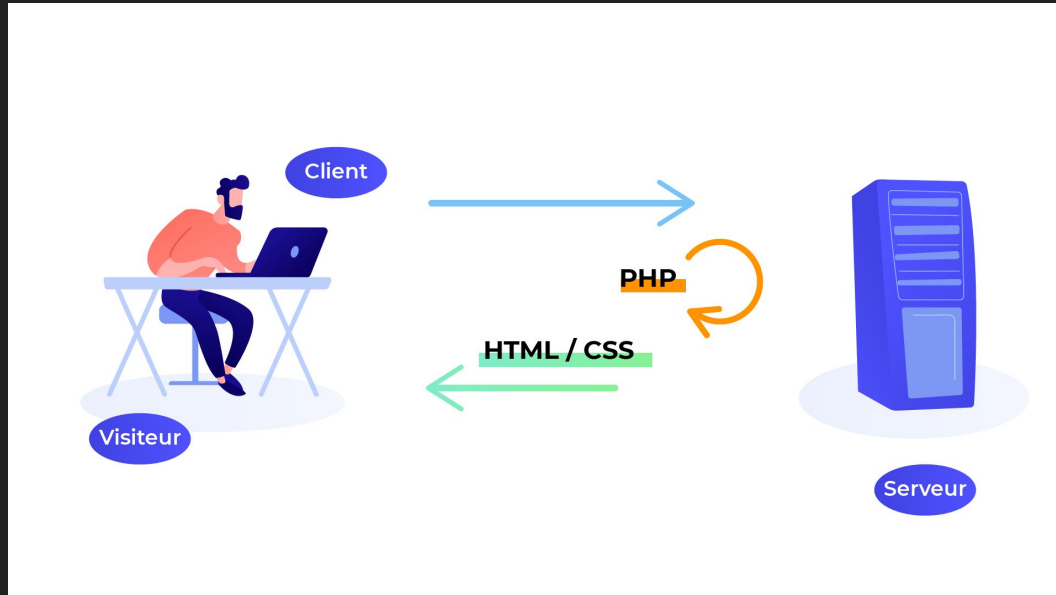
# clients / serveurs

Internet est un **réseau** composé d'ordinateurs.

Ceux-ci peuvent être classés en deux catégories :

- **Clients** : ce sont les ordinateurs des internautes comme vous. Votre ordinateur fait donc partie de la catégorie des clients. Chaque client représente un visiteur d'un site web.
- **Serveurs** : ce sont des ordinateurs puissants qui stockent et délivrent des sites web aux internautes, c'est-à-dire aux clients. La plupart des internautes n'ont jamais vu un serveur de leur vie. Pourtant, les serveurs sont indispensables au bon fonctionnement du Web.

# clients / serveurs





# PHP: Il était une fois...

Il a été créé par Rasmus Lerdorf en 1994 pour gérer son site web personnel et est depuis l'un des langages de programmation les plus populaires pour la construction de sites web dynamiques.

L'acronyme PHP signifiait à l'origine : "Personal Home Page", mais il est maintenant interprété comme "PHP: Hypertext Preprocessor" (préprocesseur d'hypertexte PHP).



# PHP: côté serveur

PHP est un **langage de script côté serveur** largement utilisé conçu pour le développement web.

Les langages côté serveur permettent de **traiter les requêtes**, d'**accéder aux bases de données**, de générer des **pages web dynamiques** et d'effectuer d'autres opérations du côté du serveur.

Ces langages permettent aux développeurs de créer des applications web puissantes et interactives en tirant parti des fonctionnalités et des capacités du serveur.





# Pourquoi apprendre PHP?

PHP est un langage interprété, ce qui signifie qu'il est exécuté sur le serveur et génère un HTML dynamique qui est renvoyé au navigateur du client.

Il peut se connecter à diverses bases de données, manipuler des fichiers, gérer des sessions utilisateur et bien plus encore.



# PHP: pourquoi le choisir ?

PHP est apprécié pour sa facilité d'apprentissage, sa flexibilité et sa vaste communauté de développeurs qui fournissent une **grande quantité de ressources et de bibliothèques**.

Il est compatible avec la plupart des serveurs web et peut être intégré dans des pages HTML ou utilisé avec des frameworks tels que **Symfony**, **Laravel** et **WordPress**.

# PHP



En somme, PHP est un langage de script côté serveur qui permet de créer des applications web dynamiques et interactives.

Il est largement utilisé dans l'industrie du développement web en raison de sa facilité d'apprentissage, de sa polyvalence et de sa compatibilité avec de nombreux outils et frameworks.



# Découverte





# Php : comment ça marche ?

Pour utiliser **PHP**, nous allons installer **XAMPP** qui fournit un environnement de développement :

**XAMPP**





# XAMPP : dans le détail

**XAMPP** est un logiciel de développement web qui fournit un environnement de serveur local pour exécuter et tester des applications web sur votre propre ordinateur. Le nom **XAMPP** est dérivé des initiales des différents composants qu'il intègre :

**X** : Cross-platform

**A** : Apache, le serveur web le plus largement utilisé.

**M** : MySQL, système de gestion de base de données relationnelle.

**P** : PHP, le langage de script côté serveur.

**P** : Perl, un langage de script polyvalent.

# XAMPP



**XAMPP** est souvent utilisé comme une solution tout-en-un pour configurer un environnement de développement local.

Il est facile à installer et fournit une interface conviviale pour gérer les services du serveur, tels que le démarrage et l'arrêt d'Apache et MySQL.

# Les bases







# PHP : convention de nommage

Les **conventions** de nommage en **PHP** sont des **bonnes pratiques** largement acceptées pour nommer les variables, les fonctions, les classes et autres éléments dans le code **PHP**.

Le **PHP** utilise par convention le **camelCase**.





# PHP : les variables

Variables : Les variables sont des conteneurs qui permettent de stocker des valeurs.

En PHP, elles sont représentées par un signe dollar (\$) suivi du nom de la variable.

```
$message = "Hello, world!"
```



# PHP : concaténation

La concaténation en PHP est le processus de fusion de chaînes de caractères. Cela permet de combiner des valeurs de chaînes de caractères différentes pour former une seule chaîne. l'opérateur de concaténation est le point (.) :

```
$nom = "John";  
$surnom = "Doe";  
$nomComplet = $nom . " " . $surnom;  
echo $nomComplet; // Affiche "John Doe"
```



# PHP : concaténation

Une autre façon de réaliser la concaténation en PHP est d'utiliser les guillemets doubles ( " ") dans lesquels les variables sont directement insérées sans utiliser l'opérateur de concaténation :

```
$nom = "John";  
$message = "Bonjour, $nom!";  
echo $message; // Affiche "Bonjour, John!"
```



# PHP : les types de données

PHP prend en charge différents types de données tels que :

- entiers (**int**)
- nombres à virgule flottante (**float**)
- chaînes de caractères (**string**)
- booléens (**bool**)
- tableaux (**array**)
- objets (**object**)



# PHP : les tableaux associatifs

En PHP, un tableau associatif est une structure de données qui permet d'**associer des clés à des valeurs**. Contrairement aux tableaux numérotés (indexés par des entiers), les tableaux associatifs utilisent des clés personnalisées pour accéder aux valeurs.

```
$personne = array(  
    "nom" => "Dupont",  
    "prenom" => "Jean"  
);  
  
echo "Je m'appelle : " . $personne["nom"] . " " . $personne["prenom"] . " ;
```



# PHP : les constantes

Les constantes en PHP sont souvent nommées en utilisant des lettres majuscules et des traits de soulignement pour séparer les mots ou peut soit utiliser `define()` ou `const` :

```
define("MESSAGE", "Bonjour, bienvenue!");  
const PI = 3.14159;  
  
echo PI; // Affiche 3.14159  
echo MESSAGE; // Affiche "Bonjour, bienvenue!"
```



# PHP : opérateurs

PHP propose une variété d'opérateurs pour effectuer :

- des opérations arithmétiques (+, -, \*, /)
- des comparaisons (==, ===, !=, <, <=, >, >=)
- des opérations logiques (&&, ||, !)
- et bien d'autres.





# PHP : les structures de contrôle

Les **structures de contrôle**, telles que :

- les boucles (**for**, **while**, **do...while**)
- instructions conditionnelles (**if**, **else**, **elseif**)

permettent de **contrôler le flux d'exécution** du programme en fonction de certaines conditions.



# PHP : les conditions

```
$note = 75;  
if ($note >= 90) {  
    echo "Très bien !";  
} elseif ($note >= 80) {  
    echo "Bien fait !";  
} elseif ($note >= 70) {  
    echo "Pas mal !";  
} else {  
    echo "Peux mieux faire.";  
}
```



# PHP : les boucles

```
for ($i = 0; $i < 5; $i++) {  
    echo "Itération : " . $i . "<br>";  
}
```



# PHP : la boucle foreach()

```
$fruits = array("pomme", "banane", "orange");
```

```
foreach ($fruits as $fruit) {  
    echo $fruit . "<br>";  
}
```



# PHP : la boucle foreach() (tableau associatif)

```
$car = ['model' => 'bwm', 'color' => 'jaune'];
```

```
foreach ($car as $key => $value) {  
    echo $key . ':' . $value . "<br>";  
}
```



# PHP : les fonctions

Les fonctions sont des blocs de code réutilisables qui effectuent une tâche spécifique.

En **PHP**, il est possible de créer des fonctions personnalisées et d'utiliser des fonctions prédéfinies pour effectuer des opérations courantes.



# PHP : `var_dump()`

La fonction `var_dump()` est principalement utilisée à des fins de débogage et d'inspection des variables.

Elle affiche des informations détaillées sur une variable, y compris son type, sa valeur et sa structure.

Elle est utile pour afficher le contenu de tableaux, d'objets et d'autres types de données complexes.

Elle affiche également la longueur des chaînes de caractères et le nombre d'éléments dans un tableau.



# PHP : manipulation de chaînes de caractères

- **substr()** : Extrait une partie d'une chaîne.
- **strtolower()** : Convertit une chaîne en minuscules.
- **strtoupper()** : Convertit une chaîne en majuscules.
- **str\_replace()** : Remplace toutes les occurrences d'une sous-chaîne dans une chaîne.
- **strlen()** : Retourne la longueur d'une chaîne de caractères.
- **strpos()** : Recherche la position d'une sous-chaîne dans une chaîne.





# PHP : manipulation des tableaux

- `count()`: Retourne le nombre d'éléments d'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 ou plusieurs tableaux.
- `array_search()`: Recherche la clé d'une valeur donnée dans un tableau.



# PHP : gestion des dates et du temps

- **date()** : Formate une date et/ou une heure selon un format spécifié.
- **time()** : Retourne le nombre de secondes écoulées depuis le 1er janvier 1970.
- **strtotime()** : Convertit une date textuelle en un timestamp Unix.
- **strtotime()** : Convertit un timestamp Unix en une date textuelle lisible.



# PHP : les superglobales

En **PHP**, les **superglobales** sont des variables prédéfinies qui sont disponibles dans tous les contextes du script. Elles sont accessibles de manière globale, ce qui signifie qu'elles peuvent être utilisées n'importe où dans le script sans avoir besoin d'être passées comme arguments de fonction ou déclarées à nouveau.

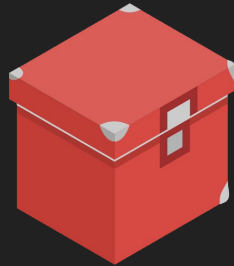
- `$_GET`
- `$_POST`
- `$_SERVER` (informations sur le serveur et la requête HTTP),
- `$_FILES`



# PHP : les inclusions de fichiers

Les inclusions de fichiers en **PHP** permettent de réutiliser du code à partir d'autres fichiers.

Elles sont utiles pour organiser le code en le divisant en plusieurs fichiers et pour éviter la duplication de code :





# PHP : include ou include\_once

L'inclusion avec **include** : **include** est utilisée pour inclure un fichier spécifié dans le script en cours, si le fichier inclus n'est pas trouvé, une simple alerte est générée et l'exécution du **script se poursuit** :

```
include 'fichier.php';  
include_once 'fichier.php';
```



# PHP : require et require\_once

La fonction **require** est similaire à **include**, mais elle **génère une erreur fatale** si le fichier inclus n'est pas trouvé.

Si le fichier inclus est essentiel pour le fonctionnement du script, utilisez **require** plutôt que **include** :

```
require 'fichier.php';  
require_once 'fichier.php';
```



# PHP : redirection

vous pouvez effectuer une **redirection** vers une autre page en utilisant la fonction **header()** avec l'en-tête "**Location**".

```
<?php
```

```
    // Redirection vers une autre page
```

```
    header("Location: chemin_vers_autre_page.php");
```

```
    exit;
```

```
    // Assurez-vous d'utiliser exit() après la redirection
```

```
?>
```



# PHP : les sessions

Les sessions en **PHP** sont un mécanisme permettant de stocker des **données persistantes entre différentes requêtes HTTP** d'un même utilisateur.

Elles sont utilisées pour maintenir l'état de l'application et permettre aux utilisateurs de rester authentifiés ou de stocker des informations spécifiques à leur session.





# PHP : créer une session

Pour utiliser les sessions, vous devez d'abord démarrer une session à l'aide de la fonction `session_start()`.

Cette fonction doit être appelée avant tout autre sortie HTML ou texte.

```
session_start();  
$nom = $_SESSION['nom'];  
$age = $_SESSION['age'];  
echo "Bonjour, $nom ! Vous avez $age ans.";
```



# PHP : créer une session

Lorsque vous avez terminé d'utiliser la session, vous pouvez la détruire en appelant la fonction `session_destroy()`. Cela supprimera toutes les données de la session en cours.

```
session_destroy();
```



# PHP : les cookies

Les **cookies** en **PHP** sont de petits fichiers de données qui sont stockés sur le navigateur de l'utilisateur. Ils sont utilisés pour **stocker des informations** spécifiques à un utilisateur et permettre la persistance des données entre les différentes requêtes **HTTP**.





# PHP : les cookies

Vous pouvez définir un cookie en utilisant la fonction `setcookie()`. Cette fonction accepte plusieurs paramètres tels que le nom du cookie, sa valeur, sa durée de vie, le chemin, le domaine, etc.

```
setcookie('nom', 'John', time() + 3600, '/');
```

Les cookies définis par le serveur peuvent être accessibles via la superglobale `$_COOKIE` :

```
$nom = $_COOKIE['nom'];  
echo "Bonjour, $nom !";
```

# Manipulation des données





# PHP : get

`$_GET` est une **superglobale** qui est utilisée pour récupérer les données envoyées via une requête HTTP GET.

Lorsqu'un formulaire HTML est soumis avec la méthode GET ou lorsqu'un utilisateur accède à une page avec des paramètres d'URL, les données sont incluses dans la partie de l'URL après le symbole "?" :

`https://www.example.com?nom=John&age=25&ville=Paris`

```
echo $_GET["nom"];
```



# PHP : post

la méthode **POST** est utilisée pour envoyer des données à un script **PHP** à partir d'un formulaire **HTML** ou d'une requête **HTTP**.

Les données envoyées via **POST** sont encapsulées dans le corps de la requête **HTTP** et ne sont pas visibles dans l'URL.

Les données envoyées via **POST** peuvent être récupérées en utilisant la superglobale **\$\_POST**.

```
echo $_post["nom"];
```



# PHP : les formulaires

Chaque champ du formulaire est identifié par son attribut name dans le code **HTML** du formulaire.

Pour récupérer une valeur spécifique, utilisez `$_POST['nom_champ']`, où "nom\_champ" correspond à l'attribut name du champ.

Exemple : Si vous avez un champ de formulaire avec name="nom", vous pouvez récupérer la valeur en utilisant `$nom = $_POST['nom'];`





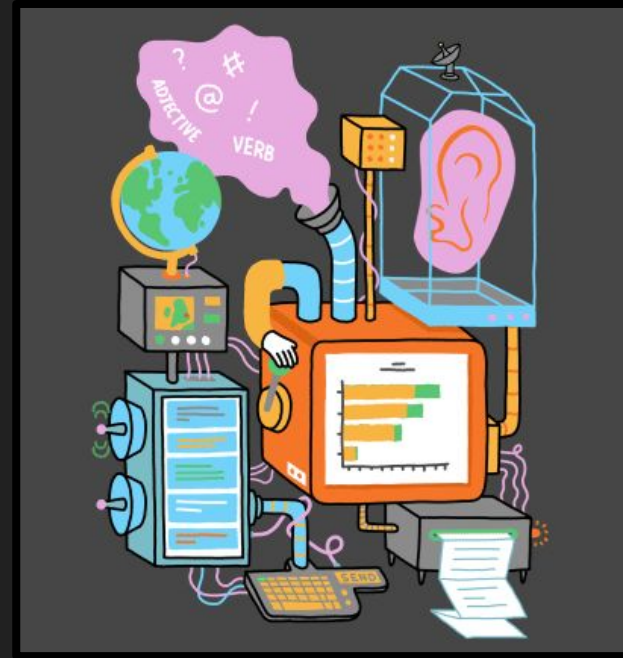
# PHP : vérifications de formulaire

Vous pouvez vérifier si un champ spécifique existe dans la requête **POST** en utilisant la fonction `isset()` :

```
if (isset($_POST['nom'])) { // code à exécuter }
```

Vous pouvez utiliser des fonctions prédéfinies de validation en PHP, telles que `filter_var()` et `preg_match()`, ou écrire vos propres fonctions de validation personnalisées.

# Base de données





# PHP : base de données

Pour se connecter à une base de données en **PHP**, vous pouvez utiliser les extensions intégrées telle **PDO** (PHP Data Objects).

Une fois que vous êtes connecté à la base de données, vous pouvez exécuter des requêtes **SQL** pour effectuer des opérations telles que la **sélection**, **l'insertion**, la **mise à jour** ou la **suppression de données**.



# PHP : connexion

```
$dsn = "mysql:host=localhost;dbname=nom_base";
$user = "utilisateur";
$password = "mot_de_passe";
try {
    $pdo = new PDO($dsn, $user, $password);
    // Configuration supplémentaire si nécessaire
} catch (PDOException $e) {
    echo "Erreur de connexion : " . $e->getMessage();
}
```



# PHP : préparer et exécuter une requête

```
$sql = "SELECT * FROM table WHERE id = :id";  
$stmt = $pdo->prepare($sql);  
$stmt->bindParam(':id', $id, PDO::PARAM_INT);  
$stmt->execute();
```



# PHP : récupérer les résultats

```
$results = $stmt->fetchAll(PDO::FETCH_ASSOC);  
foreach ($results as $row) {  
    echo $row['colonne1'];  
    echo $row['colonne2'];  
}
```

Plus loin...





# PHP : poo

Programmation Orientée Objet (POO) : est un paradigme de programmation qui permet de structurer le code de manière modulaire et de créer des objets qui représentent des entités du monde réel.

La POO offre de nombreux avantages et sert à plusieurs fins.

Apprenez à utiliser les concepts de base de la POO en PHP tels que les classes, les objets, l'encapsulation, l'héritage et le polymorphisme.





# PHP : Lectures complémentaires

Consultez certains des liens pour mieux comprendre le fonctionnement de PHP:

[PHP](#) : Documentation officielle

[Concevez votre site web](#) : Cours sur OpenClassRooms

[Apprendre PHP](#) : Tuto vidéo de Grafikart



```
echo 'MERCI';
```