Résumé PHP

Fonctionnalité	Description et Exemple
Commentaires	Les commentaires permettent d'ajouter des notes explicatives dans le code sans être exécutés. // Commentaire sur une ligne /* Commentaire sur plusieurs lignes */
Affichage de texte	echo et print sont utilisés pour afficher du texte ou des variables. echo "Bonjour le monde!"; print "Bonjour!";
Variables	Les variables stockent des données et sont précédées du symbole $. $nom = "Jean"; $age = 25;
Concaténation	La concaténation permet de combiner des chaînes de caractères ou des variables. echo "Nom: " . $nom . ", Age: " . $age;
Tableaux	Les tableaux stockent plusieurs valeurs dans une seule variable. $fruits = ["Pomme", "Banane", "Orange"]; echo $fruits[0]; // Affiche "Pomme"
Tableaux associatifs	Les tableaux associatifs utilisent des clés nommées pour accéder aux valeurs. $personne = ["nom" => "Jean", "age" => 25]; echo $personne["nom"]; // Affiche "Jean"
Conditions (if/else)	Les conditions permettent d'exécuter du code en fonction d'une expression booléenne. if ($age > 18) { echo "Majeur"; } else { echo "Mineur"; }
Boucles (for)	Les boucles for répètent un bloc de code un nombre spécifié de fois. for ($i = 0; $i < 5; $i++) { echo $i; } // Affiche 0 1 2 3 4
Boucles (foreach)	Les boucles foreach parcourent chaque élément d'un tableau. foreach ($fruits as $fruit) { echo $fruit; } // Affiche chaque fruit
Fonctions	Les fonctions encapsulent du code réutilisable. function direBonjour($nom) { return "Bonjour, " . $nom; } echo direBonjour("Jean"); // Affiche "Bonjour, Jean"
Inclusion de fichiers	include et require permettent d'inclure des fichiers PHP. require génère une erreur fatale si le fichier est manquant. include 'header.php'; require 'config.php';
Superglobales	Les superglobales sont des variables prédéfinies accessibles partout dans le script. $_GET, $_POST, $_SESSION, $_COOKIE, etc.
Traitement des formulaires	Les formulaires HTML peuvent envoyer des données via GET ou POST. **Exemple avec GET :** <form action="traitement.php" method="get"> <input type="text" name="nom"> <input type="submit">

	</form> Dans traitement.php : $nom = $_GET['nom']; echo "Bonjour, " . $nom; **Exemple avec POST :** <form action="traitement.php" method="post"> <input type="text" name="nom"> <input type="submit"> </form> Dans traitement.php : $nom = $_POST['nom']; echo "Bonjour, " . $nom;

Sessions	Les sessions stockent des données utilisateur entre les pages.
	session_start();
	$_SESSION['utilisateur'] = "Jean";
	echo $_SESSION['utilisateur']; // Affiche "Jean"

Cookies	Les cookies stockent des données sur le navigateur de l'utilisateur.
	setcookie("utilisateur", "Jean", time() + 3600);
	echo $_COOKIE['utilisateur']; // Affiche "Jean"

Manipulation de chaînes	PHP offre des fonctions pour manipuler les chaînes de caractères.
	echo strlen("Bonjour"); // Affiche 7
	echo strpos("Bonjour", "j"); // Affiche 3

Manipulation de dates	Les fonctions de date permettent de formater et manipuler des dates.
	echo date("Y-m-d H:i:s"); // Affiche la date actuelle
	echo time(); // Affiche le timestamp actuel

Gestion des erreurs	Les exceptions permettent de gérer les erreurs de manière structurée. try { // code } catch (Exception $e) { echo $e->getMessage(); }
Classes et objets	Les classes sont des modèles pour créer des objets. class Utilisateur { public $nom; } $user = new Utilisateur(); $user->nom = "Jean";
Héritage	L'héritage permet à une classe d'étendre les propriétés et méthodes d'une autre classe. class Admin extends Utilisateur { public $role = "admin"; }
Interfaces	Les interfaces définissent des contrats que les classes doivent implémenter. interface iUtilisateur { public function getNom(); }
Traits	Les traits permettent de réutiliser des méthodes dans plusieurs classes. trait Loggable { public function log($msg) { echo $msg; } }
Manipulation de fichiers	PHP permet de lire, écrire et manipuler des fichiers. $file = fopen("fichier.txt", "r"); echo fread($file, filesize("fichier.txt"));

Base de données (MySQLi)	MySQLi est une extension pour interagir avec les bases de données MySQL. $conn = new mysqli("localhost", "user", "pass", "db"); $result = $conn->query("SELECT * FROM users");
Base de données (PDO)	PDO est une abstraction pour travailler avec plusieurs types de bases de données. $pdo = new PDO("mysql:host=localhost;dbname=db", "user", "pass"); $stmt = $pdo->query("SELECT * FROM users");
Fonctions anonymes (closures)	Les fonctions anonymes sont des fonctions sans nom, souvent utilisées comme callbacks. $addition = function($a, $b) { return $a + $b; }; echo $addition(2, 3); // Affiche 5
Générateurs	Les générateurs permettent de générer des séquences de valeurs à la volée. function genereNombres() { for ($i = 0; $i < 5; $i++) { yield $i; } } foreach (genereNombres() as $nombre) { echo $nombre; }
Namespaces	Les namespaces organisent le code en espaces de noms pour éviter les conflits. namespace MonProjet; class MaClasse {}
Autoloading	L'autoloading charge automatiquement les classes sans require. spl_autoload_register(function ($class) { include $class . '.php'; });

opérateurs en PHP

Type d'opérateur	Opérateur	Description	Exemple
Arithmétiques	+	Addition	$a = 5 + 3; // $a vaut 8
	-	Soustraction	$a = 5 - 3; // $a vaut 2
	*	Multiplication	$a = 5 * 3; // $a vaut 15
	/	Division	$a = 6 / 3; // $a vaut 2
	%	Modulo (reste de la division)	$a = 5 % 3; // $a vaut 2
	**	Exponentiation (PHP 5.6+)	$a = 2 ** 3; // $a vaut 8
Affectation	=	Affectation simple	$a = 5; // $a vaut 5
	+=	Ajoute et affecte	$a += 3; // équivaut à $a = $a + 3
	-=	Soustrait et affecte	$a -= 3; // équivaut à $a = $a - 3
	*=	Multiplie et affecte	$a *= 3; // équivaut à $a = $a * 3
	/=	Divise et affecte	$a /= 3; // équivaut à $a = $a / 3

	%=	Modulo et affecte	$a %= 3; // équivaut à $a = $a % 3
	**=	Exponentiation et affecte (PHP 5.6+)	$a **= 3; // équivaut à $a = $a ** 3
	.=	Concatène et affecte	$a = "Bon"; $a .= "jour"; // $a vaut "Bonjour"
Comparaison	==	Égalité (valeur)	5 == "5"; // true
	===	Égalité stricte (valeur et type)	5 === "5"; // false
	!= ou <>	Différence (valeur)	5 != "5"; // false
	!==	Différence stricte (valeur ou type)	5 !== "5"; // true
	<	Inférieur à	5 < 10; // true
	>	Supérieur à	5 > 10; // false
	<=	Inférieur ou égal à	5 <= 5; // true
	>=	Supérieur ou égal à	5 >= 10; // false

	<=>	Opérateur de comparaison combiné (PHP 7+)	5 <=> 3; // 1 (5 > 3)
			5 <=> 5; // 0 (5 == 5)
			5 <=> 7; // -1 (5 < 7)

Logiques	&& ou and	ET logique	true && false; // false
Concaténation	.	Concatène deux chaînes	$a = "Bon" . "jour"; // $a vaut "Bonjour"
Incrémentation /Décrémentation	++	Incrémente une variable de 1	$a = 5; $a++; // $a vaut 6
	--	Décrémente une variable de 1	$a = 5; $a--; // $a vaut 4
Ternaire	?:	Opérateur ternaire (condition ? valeur_si_vrai : valeur_si_faux)	$a = (5 > 3) ? "Vrai" : "Faux"; // $a vaut "Vrai"
Type	instanceof	Vérifie si un objet est une instance d'une classe	$a = new MaClasse(); $a instanceof MaClasse; // true

Explications supplémentaires :

1. **Opérateurs arithmétiques** : Utilisés pour les calculs mathématiques de base.
2. **Opérateurs d'affectation** : Permettent d'assigner des valeurs à des variables, souvent combinés avec d'autres opérations.
3. **Opérateurs de comparaison** : Comparer des valeurs ou des types de données.
4. **Opérateurs logiques** : Utilisés pour combiner des conditions.
5. **Opérateur de concaténation** : Combine des chaînes de caractères.

6. **Opérateurs d'incrémentation/décrémentation** : Augmentent ou diminuent une valeur de 1.
7. **Opérateur ternaire** : Permet de simplifier des conditions if-else.
8. **Opérateur d'exécution** : Exécute des commandes système.
9. **Opérateur instanceof** : Vérifie le type d'un objet.

Manipulation des formulaires avec GET et POST :

GET

- La méthode GET envoie les données du formulaire via l'URL.
- Les données sont visibles dans l'URL et limitées en taille.
- Utilisé pour des requêtes non sensibles (comme des recherches).

Exemple :

Dans index.html

```
<form action="traitement.php" method="get">
  <input type="text" name="nom" placeholder="Votre nom">
  <input type="submit" value="Envoyer">
</form>
```

Dans traitement.php :

```
$nom = $_GET['nom']; // Récupère la valeur du champ "nom"
echo "Bonjour, " . htmlspecialchars($nom); // Sécurise l'affichage
```

POST

- La méthode POST envoie les données du formulaire dans le corps de la requête HTTP.
- Les données ne sont pas visibles dans l'URL et peuvent être plus volumineuses.
- Utilisé pour des données sensibles (comme les mots de passe).

Exemple :

Dans index.html

```
<form action="traitement.php" method="post">
  <input type="text" name="nom" placeholder="Votre nom">
  <input type="submit" value="Envoyer">
</form>
```

Dans traitement.php :

```
$nom = $_POST['nom']; // Récupère la valeur du champ "nom"
echo "Bonjour, " . htmlspecialchars($nom); // Sécurise l'affichage
```