PHP

PHP, ou "PHP: Hypertext Preprocessor", est un langage de script côté serveur créé en 1994 et largement utilisé pour le développement web. Il permet de créer des pages web dynamiques qui peuvent interagir avec des bases de données et des fichiers sur le serveur. PHP est particulièrement célèbre pour sa facilité d'intégration avec le HTML et son vaste écosystème de frameworks et bibliothèques.

Points Clés de PHP Adaptés au Développement Web Côté Serveur

- **Intégration facile avec HTML**: PHP peut être intégré directement dans le code HTML, ce qui simplifie la création de contenus web dynamiques.
- **Large base de bibliothèques et de frameworks**: PHP bénéficie de nombreux frameworks (Laravel, Symfony, etc.) qui accélèrent le développement et facilitent la maintenance.
- **Gestion de base de données**: PHP supporte une multitude de systèmes de gestion de base de données comme MySQL, PostgreSQL et SQLite, rendant la gestion de données facile et flexible.
- **Hébergement répandu**: La majorité des fournisseurs d'hébergement web supportent PHP, souvent sans frais supplémentaires.
- **Grande communauté**: PHP dispose d'une large communauté de développeurs, ce qui assure une grande quantité de ressources, de tutoriels et de forums pour aider les nouveaux développeurs.
- **Traitement côté serveur**: Comme PHP s'exécute sur le serveur, cela permet un contrôle complet du comportement de l'application avant d'envoyer le contenu au client, optimisant ainsi la sécurité et l'accès aux ressources du serveur.
- **Flexible et dynamique**: PHP est adapté pour créer des applications web petites à grandes échelles, et peut gérer des sites statiques aussi bien que des applications web complexes.
- **Support pour les API**: PHP peut facilement travailler avec des API et des services web, facilitant l'intégration avec d'autres applications et plateformes.

Utilisation de PHP comme Langage de Script Local

En plus de son rôle prédominant dans le développement web, PHP peut également être utilisé localement comme un langage de script pour des tâches de programmation générale via la ligne de commande (CLI). Cette approche offre plusieurs avantages pour l'automatisation des tâches et la gestion de systèmes :

- **Exécution en ligne de commande**: PHP en mode CLI permet d'écrire des scripts qui automatisent des tâches systèmes, gèrent des fichiers, et effectuent des opérations de traitement de données sans nécessiter un serveur web.
- **Syntaxe familière et fonctionnalités riches**: Les développeurs peuvent utiliser la même syntaxe et les mêmes fonctions que pour le développement web, ce qui rend la transition vers des scripts CLI plus facile.
- **Bibliothèques et extensions robustes**: PHP dispose de nombreuses extensions qui facilitent la manipulation de divers formats de données et l'intégration avec d'autres applications.
- **Utilisation pratique pour l'automatisation**: Les scripts PHP CLI peuvent automatiser des tâches comme les sauvegardes, l'analyse de logs, et l'envoi automatique de courriels.

Cette polyvalence fait de PHP un outil utile non seulement pour le développement web, mais aussi pour le scripting local, étendant ainsi son applicabilité à une variété de scénarios en dehors du web.

Exemples de scripts en PHP

1. Balises PHP

PHP peut être intégré dans du HTML en utilisant les balises PHP :

```php
<?php
  echo "Hello, World!";
?>
```

Vous pouvez également utiliser les balises de raccourci (si elles sont activées) :

```php
<?= "Hello, World!"; ?>
```

2. Variables et Types de Données

En PHP, les variables sont déclarées avec un signe dollar `$` suivi du nom de la variable. PHP est un langage à typage dynamique, donc vous n'avez pas besoin de déclarer le type de la variable.

```php
<?php
  $text = "Hello, World!";  // String
  $number = 100;            // Integer
  $float = 10.50;           // Floating point number
  $bool = true;             // Boolean
  $array = [1, 2, 3];       // Array
?>
```

3. Structures de Contrôle

PHP supporte les structures de contrôle classiques comme les conditionnelles et les boucles.

Conditionnelles :

```php
<?php
  $age = 20;

  if ($age >= 18) {
    echo "Adult";
  } else {
    echo "Minor";
  }
?>
```

Boucles :

```php
<?php
  // Boucle for
  for ($i = 0; $i < 5; $i++) {
    echo $i;
  }

  // Boucle while
  $j = 0;
  while ($j < 5) {
    echo $j;
    $j++;
  }
?>
```

4. Fonctions

Les fonctions en PHP sont définies avec le mot-clé `function` :

```php
<?php
  function sayHello($name) {
    return "Hello, " . $name . "!";
  }

  echo sayHello("Alice");
?>
```

5. Traitement d'un formulaire

```php
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Nom: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = htmlspecialchars($_REQUEST['fname']);
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo "Name is: " . $name;
    }
}
?>
```

Ce formulaire HTML envoie des données à la même page PHP, qui traite ensuite les données et affiche le nom saisi par l'utilisateur.

6. Programmation Orientée Objet

PHP supporte la programmation orientée objet. Vous pouvez définir des classes, des propriétés et des méthodes.

```php
<?php
  class Person {
    public $name;
    public $age;

    public function __construct($name, $age) {
      $this->name = $name;
      $this->age = $age;
    }

    public function greet() {
      return "Hello, my name is " . $this->name;
    }
  }

  $bob = new Person("Bob", 25);
  echo $bob->greet();
?>
```

7. Gestion des Sessions

Les sessions sont utilisées pour sauvegarder des informations à travers plusieurs pages.

```php
<?php
  session_start();  // Démarre la session

  $_SESSION['user'] = "Alice";  // Stocke une info dans la session

  echo "The user is " . $_SESSION['user'];
?>
```

Ce code initialise une session et stocke une information au sein de cette session, qui peut être utilisée sur différentes pages du même site.

8. Connexion à une Base de Données

Le code suivant illustre comment se connecter à une base de données MySQL avec PDO (PHP Data Objects) :

```php
<?php
  $host = '127.0.0.1';
  $db   = 'test_db';
  $user = 'root';
```

```php
    $pass = 'password';
    $charset = 'utf8mb4';

    $dsn = "mysql:host=$host;dbname=$db;charset=$charset";
    $options = [
      PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
      PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
      PDO::ATTR_EMULATE_PREPARES   => false,
    ];

    try {
      $pdo = new PDO($dsn, $user, $pass, $options);
      echo "Connected successfully";
    } catch (\PDOException $e) {
      throw new \PDOException($e->getMessage(), (int)$e->getCode());
    }
  ?>
```

Ce script PHP crée une connexion à une base de données MySQL et vérifie si la connexion est réussie.