FORMATION EN PHP

PLAN DE FORMATION SUR LES BASES DU LANGAGE PHP :

Introduction à PHP :

Avantages de PHP par rapport à d'autres langages de programmation

Installation et configuration de l'environnement de développement PHP

Syntaxe de base de PHP :

Variables et types de données

Opérateurs et expressions

Structures de contrôle (if/else, switch/case, boucles)

Fonctions et tableaux

Gestion des formulaires HTML avec PHP :

Collecte de données de formulaire

Validation de données de formulaire

Utilisation des superglobales

Utilisation de MySQL avec PHP :

Introduction à MySQL

Connexion à une base de données MySQL avec PHP

Requêtes SQL avec PHP

Affichage des résultats de la requête

INTRODUCTION

PHP (Hypertext Preprocessor) est un langage de programmation open-source, principalementutilisépour le développement d'applications web côté serveur. Il est conçu pour être facile à apprendre et à utiliser, et est compatible avec la plupart des serveurs web, des systèmes d'exploitation et des bases dedonnées.

C'est un langage de script côté serveur, ce qui signifie que le code PHP est exécuté sur le serveur avant d'être envoyé au navigateur web. Cela permet aux développeurs de créer des pages web dynamiquesqui peuvent être personnalisées en fonction des entrées de l'utilisateur et de fournir une expérience utilisateur plus interactive.

Il est souvent utilisé en conjonction avec des bases de données relationnelles telles que MySQL, pour stocker et récupérer des données. Il peut également être utilisé pour créer des applications web complexes telles que des systèmes de gestion de contenu (CMS), des forums en ligne, des paniersd'achat en ligne et bien plus encore.

C'est un langage très populaire et est utilisé par des millions de sites web dans le mond e entier.Ilestfacile à apprendre pour les débutants et offre également une grande flexibilité pour les développeurs expérimentés.

AVANTAGE

Voici quelques avantages de PHP par rapport à d'autres langages de programmation :

- Gratuit et open-source : PHP est un langage de programmation gratuit et open-source, ce qui signifie que le code source est disponible gratuitement et peut être modifié etdistribuélibrement.
- Facile à apprendre : PHP est un langage de programmation facile à apprendre pour les débutants, avec une syntaxe simple et des fonctionnalités intuitives.
- Large communauté : PHP a une grande communauté de développeurs actifs qui partagent leurs connaissances et leur expérience, et qui créent des bibliothèques de code open-sourcepouraider les autres développeurs.
- Grande compatibilité : PHP est compatible avec la plupart des serveurs web, des systèmes d'exploitation et des bases de données, ce qui le rend facile à intégrer dans des environnements existants.
- Performances élevées : PHP est un langage de script côté serveur rapide et efficace, qui peut traiter un grand nombre de requêtes simultanément.
- Grande flexibilité : PHP est un langage de programmation très flexible qui peut être utilisé pour une variété d'applications web, allant des sites web statiques aux applications web complexes.

- Évolutif : PHP est un langage de programmation évolutif qui peut être utilisé pour créer des applications web de toutes tailles, des petites applications à une seule page aux grandes applications d'entreprise.
- En somme, PHP est un langage de programmation populaire et polyvalent qui offre de nombreux avantages pour les développeurs web.

INSTALLATION :

Voici quelques étapes générales pour installer et configurer PHP sur un système d'exploitation Windows:

- Telecharger et installer un serveur local (WAMP, XAMPP ou LARAGON)
- Installer un editeur de texte (VSCode)
- Aller dans le dossier htdocs (Xampp) ou WWW (WAMP et LARAGON) et créer votre projet
- Ouvrer votre projet avec votre editeur de code pour créer votre premier fichier php portant l'extention **.php**

NB : le php s'ecrit dans la balise < ?php … ?>

VARIABLE ET TYPE DE DONNEES

En PHP, les variables sont des éléments qui permettent de stocker des données temporaires ou permanentes. Les variables en PHP sont précédées du symbole dollar ($), suivi du nom de la variable. Voici un exemple de déclaration de variable en PHP :

$nom_variable = valeur;

Les types de données en PHP incluent :

Les chaînes de caractères (string) : les chaînes de caractères sont des séquences de caractères, délimitées par des guillemets simples ou doubles.

Les nombres entiers (integer) : les nombres entiers sont des nombres sans décimales.

Les nombres à virgule flottante (float) : les nombres à virgule flottante sont des nombres avec une partie décimale.

Les booléens (boolean) : les booléens sont des valeurs logiques qui peuvent être vraies (true) ou fausses (false).

Les tableaux (array) : les tableaux sont des collections de valeurs qui peuvent être de différents types de données.

Les objets (object) : les objets sont des instances de classes, qui peuvent avoir leurs propres propriétéset méthodes.

Les ressources (resource) : les ressources sont des références à des ressources externes, telles que des fichiers ou des connexions de base de données.

Il est important de comprendre les types de données en PHP car cela permet de manipuler les variables correctement et d'effectuer des opérations appropriées sur ces variables.

OPERATEUR ET EXPRESSION

En PHP, les opérateurs sont des symboles qui permettent d'effectuer des opérations sur des variableset des valeurs. Voici quelques exemples d'opérateurs en PHP :

- Les opérateurs arithmétiques :

- + (addition),
- - (soustraction),
- * (multiplication),
- / (division),
- % (modulo).

- Les opérateurs d'affectation :

- = (affectation simple),
- += (addition et affectation),
- -= (soustraction et affectation),
- *= (multiplication et affectation),
- /= (division et affectation),
- %= (modulo et affectation).

- Les opérateurs de comparaison :

- == (égal à),
- != (différent de),
- < (inférieur à),
- > (supérieur à),
- <= (inférieur ou égal à),
- >= (supérieur ou égal à).

- Les opérateurs logiques :

- && (ET logique),
- || (OU logique),
- ! (NON logique).

- Les opérateurs d'incrémentation et de décrémentation :

- ++ (incrémentation),
- -- (décrémentation).

Les expressions en PHP sont des combinaisons de variables, de valeurs et d'opérateurs qui permettent d'effectuer des calculs et des opérations. Voici quelques exemples d'expressions en PHP :

```
$a = 5;

$b = 3;

$c = $a + $b; // l'expression $a + $b donne la valeur 8, qui est affectée à la variable $c

$d = ($a * $b) % 2; // l'expression ($a * $b) % 2 donne la valeur 1, qui est affectée à la variable $d

$e = ($a > $b) && ($c < 10); // l'expression ($a > $b) && ($c < 10) donne la valeur true, qui est affectée à la variable $e

$f = ++$a; // l'expression ++$a incrémente la variable $a de 1 et affecte la nouvelle valeur 6 à la variable $f
```

En utilisant les opérateurs et les expressions en PHP, vous pouvez effectuer des calculs, des comparaisons et des opérations logiques pour créer des applications web dynamiques et interactives.

STRUCTURE DE CONTROLE

En PHP, les structures de contrôle permettent d'exécuter des blocs de code en fonction de certaines conditions ou de manière répétitive. Voici les principales structures de contrôle en PHP :

La structure if/else : permet d'exécuter un bloc de code si une condition est vraie, sinon exécute un autre bloc de code.

```php
<?php
    if ($condition) {
        // code à exécuter si la condition est vraie
    } else {
        // code à exécuter si la condition est fausse
    }
?>
```

La structure switch/case : permet d'exécuter différents blocs de code en fonction de la valeur d'une variable.

```php
switch ($variable) {
    case valeur1:
        // code à exécuter si la variable a la valeur 1
        break;
    case valeur2:
        // code à exécuter si la variable a la valeur 2
        break;
    default:
        // code à exécuter si la variable n'a aucune des valeurs précédentes
}
```

Les boucles : permettent d'exécuter un bloc de code de manière répétitive.

La boucle while : exécute un bloc de code tant qu'une condition est vraie.

```php
while (condition) {
    // code à exécuter tant que la condition est vraie
}
```

La boucle do/while : exécute un bloc de code au moins une fois, puis tant qu'une condition est vraie.

```php
do {
    // code à exécuter au moins une fois
} while (condition);
```

La boucle for : exécute un bloc de code un nombre déterminé de fois.

```php
for (initialisation; condition; incrémentation) {
    // code à exécuter pour chaque itération de la boucle
}
```

La boucle foreach : permet d'itérer sur les éléments d'un tableau.

```php
foreach ($tableau as $valeur) {
  // code à exécuter pour chaque élément du tableau
}
```

En utilisant ces structures de contrôle, il est possible d'écrire des programmes PHP qui effectuent des actions conditionnelles et répétitives, ce qui permet d'automatiser des tâches et d'optimiser les performances.

EXEMPLE :

Bien sûr, voici quelques exemples d'utilisation des structures de contrôle en PHP :

1. Exemple d'utilisation de la structure if/else :

```
$age = 25;

if ($age >= 18) {
  echo "Vous êtes majeur.";
} else {
  echo "Vous êtes mineur.";
}
```

Dans cet exemple, la structure if/else permet de vérifier si la variable $age est supérieure ou égale à 18. Si c'est le cas, le message "Vous êtes majeur." est affiché, sinon le message "Vous êtes mineur."estaffiché.

2. Exemple d'utilisation de la structure switch/case :

```
$fruit = "pomme";

switch ($fruit) {
  case "pomme":
    echo "Le fruit est une pomme.";
    break;
```

```
  case "orange":
    echo "Le fruit est une orange.";
    break;
  default:
    echo "Le fruit n'est ni une pomme ni une orange.";
}
```

Dans cet exemple, la structure switch/case permet de vérifier la valeur de la variable $fruit. Si elle vaut "pomme", le message "Le fruit est une pomme." est affiché, si elle vaut "orange", le message "Le fruitest une orange." est affiché, sinon le message "Le fruit n'est ni une pomme ni une orange." est affiché.

3. Exemple d'utilisation de la boucle for :

```
for ($i = 1; $i <= 10; $i++) {
  echo $i . "<br>";
}
```

Dans cet exemple, la boucle for permet d'afficher les nombres de 1 à 10 en utilisant la variable $i. La boucle commence par initialiser la variable $i à 1, puis exécute le bloc de code tant que la condition $i <= 10 est vraie, et incrémente la variable $i de 1 à chaque itération.

4. Exemple d'utilisation de la boucle foreach :

```
$prenoms = array("Alice", "Bob", "Charlie", "David");

foreach ($prenoms as $prenom) {
  echo $prenom . "<br>";
}
```

Dans cet exemple, la boucle foreach permet d'afficher les éléments du tableau $prenoms en utilisant la variable $prenom. La boucle itère sur chaque élément du tableau et affecte sa valeur à la variable $prenom, puis exécute le bloc de code pour chaque élément.

En PHP, les fonctions permettent d'encapsuler un bloc de code qui peut être exécuté plusieurs fois avec différentes entrées et sorties. Les tableaux, quant à eux, permettent de stocker des collections de données dans une seule variable. Voici quelques exemples d'utilisation des fonctions et des tableaux en PHP :

1. Exemple de fonction :

```php
function addition($a, $b) {
  return $a + $b;
}

$resultat = addition(3, 5); // appelle la fonction addition avec les arguments 3 et 5

echo $resultat; // affiche 8
```

Dans cet exemple, la fonction addition prend deux paramètres $a et $b, effectue l'opération d'addition et retourne le résultat. La fonction est ensuite appelée avec les arguments 3 et 5, et le résultat est stocké dans la variable $resultat.

2. Exemple de tableau :

```php
$prenoms = array("Alice", "Bob", "Charlie", "David");

echo $prenoms[1]; // affiche "Bob"
```

Dans cet exemple, un tableau de prénoms est créé en utilisant la fonction array(). Chaque élément du tableau est indexé à partir de zéro, donc le deuxième élément du tableau (index 1) est "Bob". Le prénom "Bob" est ensuite affiché en utilisant l'index du tableau.

3. Exemple d'utilisation de la fonction count() avec un tableau :

```php
$prenoms = array("Alice", "Bob", "Charlie", "David");

$nombre_prenoms = count($prenoms);

echo "Il y a " . $nombre_prenoms . " prénoms dans le tableau.";
```

Dans cet exemple, la fonction count() est utilisée pour compter le nombre d'éléments dans le tableau $prenoms. Le nombre de prénoms est ensuite affiché en utilisant la fonction echo().

COLLECTE DE DONNEES DU FORMULAIRE

Voici un exemple plus détaillé de collecte de données de formulaire en PHP :

1. Créez un formulaire HTML avec les champs que vous souhaitez collecter. Par exemple, voici un formulaire avec un champ de texte pour le nom et un bouton de soumission :

```html
<form method="post" action="traitement.php">
  <label for="nom">Nom :</label>
  <input type="text" id="nom" name="nom">
  <br>
  <input type="submit" value="Envoyer">
</form>
```

2. Dans le fichier "traitement.php", utilisez la superglobale $_POST pour récupérer les données du formulaire. Par exemple, pour récupérer le nom entré dans le champ de texte, vous pouvez utiliser la syntaxe suivante :

```php
$nom = $_POST["nom"];
```

3. Il est important de valider et de nettoyer les données avant de les utiliser dans votre application. Par exemple, vous pouvez vérifier si le champ de texte est vide avant de l'utiliser :

```php
if (!empty($_POST["nom"])) {
  $nom = $_POST["nom"];
  echo "Bonjour " . $nom . " !";
} else {
  echo "Veuillez entrer un nom.";
}
```

Dans cet exemple, la fonction empty() est utilisée pour vérifier si le champ de texte est vide. Si c'est le cas, un message d'erreur est affiché. Sinon, le nom entré dans le champ de texte est affiché dans un message de salutation.

4. Vous pouvez également utiliser des fonctions de nettoyage pour supprimer les caractères spéciaux et les balises HTML des données du formulaire. Par exemple, la fonction strip_tags() supprime toutes les balises HTML d'une chaîne de caractères :

```php
$nom = strip_tags($_POST["nom"]);
```

```

Dans cet exemple, la fonction strip_tags() est utilisée pour supprimer toutes les balises HTML du nom entré dans le champ de texte.

En utilisant ces étapes, vous pouvez collecter et traiter les données d'un formulaire HTML en PHP.

La validation des données de formulaire est une étape importante pour assurer la sécurité et la fiabilité de votre application. Voici quelques étapes à suivre pour valider les données de formulaire en PHP :

## VALIDATION DU FORMULAIRE

1. Vérifiez que le formulaire a été soumis en utilisant la méthode POST. Vous pouvez utiliser la fonction isset() pour vérifier si la variable $_POST existe :

```php
if (isset($_POST["nom"])) {
  // Le formulaire a été soumis
} else {
  // Le formulaire n'a pas été soumis
}
```

2. Vérifiez que les champs obligatoires ont été remplis. Pour cela, vous pouvez utiliser la fonctionempty() pour vérifier si les variables $_POST sont vides :

```php
if (empty($_POST["nom"])) {
  $erreurs[] = "Le champ nom est obligatoire.";
}
```
```

```

Dans cet exemple, la variable $erreurs est initialisée comme un tableau vide. Si le champ de texte "nom" est vide, un message d'erreur est ajouté au tableau $erreurs.

3. Vérifiez que les données entrées sont du bon type. Par exemple, si vous attendez une adresse email, vous pouvez utiliser la fonction filter_var() pour valider l'adresse email :

```php
$email = $_POST["email"];

if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
  $erreurs[] = "L'adresse email n'est pas valide.";
}
```

Dans cet exemple, la fonction filter_var() est utilisée pour valider l'adresse email entrée dans le champ de texte "email". Si l'adresse email n'est pas valide, un message d'erreur est ajouté au tableau $erreurs.

4. Vérifiez que les données entrées ne contiennent pas de caractères spéciaux ou de code malveillant. Pour cela, vous pouvez utiliser la fonction htmlspecialchars() pour convertir les caractères spéciaux en entités HTML :

```php
$nom = $_POST["nom"];
$nom = htmlspecialchars($nom);
```

Dans cet exemple, la fonction htmlspecialchars() est utilisée pour convertir les caractères spéciaux du nom entré dans le champ de texte "nom" en entités HTML.
```

5. Affichez les messages d'erreur ou enregistrez les données dans votre application si aucune erreur n'a été détectée. Par exemple, vous pouvez utiliser une boucle foreach pour parcourir le tableau $erreurset afficher chaque message d'erreur :

```php
if (count($erreurs) > 0) {
  // Il y a des erreurs, affichez-les
  foreach ($erreurs as $erreur) {
    echo $erreur . "<br>";
  }
} else {
  // Il n'y a pas d'erreur, enregistrez les données dans votre application
}
```

UTILISATIONS DES VARIABLES SUPER GLOBALE

Les superglobales sont des variables prédéfinies en PHP qui sont disponibles dans tous les contextes de votre application. Elles permettent d'accéder à des informations importantes telles que les données de formulaire, les en-têtes HTTP et les variables d'environnement. Voici quelques exemples d'utilisationdes superglobales en PHP :

1. $_POST : Cette superglobale est utilisée pour récupérer les données envoyées par un formulaireHTML avecla méthode POST. Par exemple, pour récupérer le nom entré dans un champ de texte nommé"nom", vous pouvez utiliser la syntaxe suivante :

```php
$nom = $_POST["nom"];
```

2. $_GET : Cette superglobale est utilisée pour récupérer les données envoyées par un formulaire HTML avecla méthode GET ou par l'URL. Par exemple, pour récupérer la valeur du paramètre "page" dans l'URL "http://localhost/index.php?page=accueil", vous pouvez utiliser la syntaxe suivante :

```php
$page = $_GET["page"];
```

3. $_SESSION : Cette superglobale est utilisée pour stocker des données de session entre les page s de votre application. Par exemple, pour stocker le nom d'utilisateur d'un utilisateur connecté, vous pouvez utiliser la syntaxe suivante :

```php
session_start();

$_SESSION["utilisateur"] = "JohnDoe";
```

4. $_COOKIE : Cette superglobale est utilisée pour stocker des données dans un cookie qui peut être lu par le navigateur client. Par exemple, pour stocker le nom d'utilisateur dans un cookie nommé "utilisateur", vous pouvez utiliser la syntaxe suivante :

```php
setcookie("utilisateur", "JohnDoe", time() + 3600, "/");
```

5. $_SERVER : Cette superglobale est utilisée pour accéder aux informations sur le serveur telles que les en-têtes HTTP et les variables d'environnement. Par exemple, pour récupérer l'adresse IP du client,vous pouvez utiliser la syntaxe suivante :

```php
$adresse_ip = $_SERVER["REMOTE_ADDR"];
```

En utilisant les superglobales en PHP, vous pouvez accéder à des informations importantes sur votre application et vos utilisateurs. Il est important de comprendre comment utiliser ces superglobales de manière sécurisée et fiable pour éviter les failles de sécurité et les erreurs de programmation.