GALIMATECH
L'EXPERIENCE DIGITALE

Apprendre le PHP

***

Plan du cours

- Introduction
- Installation
- La Syntaxe
- Les variables
- Les Arrays
- Les Hash
- Les Fonctions
- Les Conditions
- Les Comparaisons
- Le Switch Case
- Les Boucles
- Include

Php est un langage interprété qui tourne côté serveur. Il utilise un typage dynamique et faible.
Il va nous permettre de dynamiser le contenu de notre site en communiquant avec un serveur HTTP.

Installation

Sur ce cours nous allons utiliser un système Linux de distribution Debian. Pour utiliser le PHP dans un serveur Debian nous allons installer le php-cli avec cette commande:

sudo apt update
sudo apt install php libapache2-mod-php

Nous pouvons ensuite vérifier notre version avec la commande php -v.

La syntax

Pour écrire du PHP nous allons d'abord ouvrir une balise Php comme suite:

```
<?php
```

Cette balise se ferme ensuite comme suite

```
?>
```

Cela veut donc dire tout le code entre `<?php` et `?>` est du code php

Les commentaires

Les commentaires vont permettre d'ajouter des instructions ou des explications sur le code. Ils sont utilisés pour aider à se souvenir du fonctionnement d'un bloc d'instructions ou pour aider à comprendre le code.

Nous avons deux manières d'écrire les commentaires en PHP:

- le commentaire sur une ligne avec:

```
// toute ce qui est sur cette ligne est un commentaire
```

```
# toute ce qui est sur cette ligne est aussi un commentaire
```

- le commentaire en multi ligne:

```
/*
tout ce qui est entre les signes /* et */
sera commenté
*/
```

Les variables

Les variables permettent d'enregistrer la valeur d'une donnée et son type.
Les variables sont déclarées avec le signe $ suivi du nom de la variable.

```
$nom_variable_1 = "la valeur";
```

Le nom doit être constitué de caractères alphanumériques et du underscore. Il est déconseillé d'utiliser les caractères spéciaux dans nos variables.
Php utilise un typage faible et dynamique. Cela veut dire que le type dépend de la valeur appliquée et qu'une variable peut changer de type en cours d'exécution.

En php nous avons les types suivants:

- Les chaînes de caractères aussi appelée string. Ils sont entre des quotes ou des doubles quotes.
- Les nombres entiers appelés integer ou int
- Les floats pour les nombres réels
- Les arrays qui sont des tableaux pouvant prendre tout type de données
- Le booléen qui peut avoir comme valeur true ou false.

Les arrays

Les arrays vont permettre d'enregistrer une liste de valeurs.
Ils peuvent prendre n'importe quel type de variable.

Pour créer un tableau nous utilisons la syntaxe.

```php
$tableau = [ "valeur", 'valeu', 1232, 1.4, [1,2,3]];
```

Pour récupérer une valeur dans le tableau nous allons utiliser des clés qui correspondent aux positions des éléments. En prenant en compte que le premier element est a la position zéro, le second à la position 1 ...

```php
echo $tableau[0];
```

Nous pouvons aussi utiliser la même méthode pour ajouter une nouvelle clé ou changer une valeur.

```php
echo $tableau[7] = "nouvelle valeur";
```

Les Tableaux Associatifs

Les tableaux associatifs sont des arrays dont les clés sont des références. On les appelle aussi des “hashs map”.
Pour créer un tableau associatif nous allons utiliser la syntaxe suivante:

```
$user = [
    "nom"        => "Fall",
    "prenom"     => "Soxna",
    "age"        => 50,
    "taille"     => 1.2,
    "options"    => ["option1","option2","option3","option4"],
]
```

Pour récupérer une valeur de ce tableau nous pouvons faire:

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

Les fonctions

Les fonctions sont des bouts de code qui permettent d'accomplir une tâche précise. Elles contiennent une ou plusieurs actions qui sont exécutées à chaque fois que la fonction est appelée. Et permettent ainsi d'éviter la répétition.

Pour déclarer une fonction en php on fait:

`function nomFonction(){`
`    //instructions`
`}`
	`function nomFonction($parameters){`
	`    //instructions`
	`}`

En fonction des instructions, nous pouvons ajouter des paramètres ou pas. Pour utiliser la fonction nous pouvons faire:

`functionName();`	`nomFonction($parameters);`

Les conditions

En écrivant notre code il va arriver qu'on ait besoin d'exécuter une ou plusieurs actions seulement si certaines situations sont réunies.

Nous pouvons faire cela en utilisant les conditions qui s'écrivent avec le *if statement*

```
if(conditions){
    # Instructions...
}
```

Les instructions ne seront exécutées que si la condition est vraie (true). On peut ajouter une autre condition qui sera exécutée si la première n'est pas vérifiée avec *elseif(conditions){}*.Enfin, une instruction qui s'exécute si aucune condition n'est vérifiée avec *else{}*.

```
if(condition){
    # Instructions...
} elseif (condition) {
    # instructions...
}else {
    # instructions...
}
```

Les opérateurs de comparaison

Les opérateurs de comparaison permettent, comme leur nom l'indique, de comparer deux valeurs. Comme opérateurs de comparaison nous avons:

L’opérateur d'égalité: a == b qui retourne true si ‘a’ et ‘b’ ont la même valeur.

L’opérateur de différence: a != b qui retourne true si ‘a’ et ‘b’ sont différents.

L'opérateur de supériorité: a > b vérifie si la valeur de ‘a’ est supérieur à la valeur de ‘b’.

L’opérateur d'infériorité: a < b vérifie si ‘a’ inférieure a ‘b’

L'opérateur d'égalité forte: a===b comme l'opérateur d'égalité elle vérifie si deux éléments ont la même valeur. Mais elle vérifie aussi si ces éléments sont du même type de données.

Le Switch

Le switch est une manière d'écrire des conditions plus rapidement. Il permet de remplacer une succession de if statements.

```
switch($laValeurAComparer){
    case 1:
        //instructions si c'est égal à 1
        break;
    case 2:
        //instructions si c'est égal à 2
        break;
    default:
        //instructions à exécuter si aucune des valeurs précédentes n'est vérifiée
}
```

Les boucles

Les boucles vont permettre de répéter l'exécution d'une ou de plusieurs actions.

Nous avons trois types de boucles.

La boucle while: elle s'exécute tant qu'une certaine condition est vérifiée.

```
while(condition){//instructions }
```

La boucle do...while: elle exécute le code, puis vérifie si la condition est vérifiée.

```
do{//instructions }while(conditions)
```

La boucle for:va exécuter les instructions tant que la condition est vérifiée.

```
for($i = 0; $i < $max; $i++){

    //les actions à exécuter tant que $i est inferieur à $max
}
```

Les boucles: le foreach

Le foreach est le plus souvent utilisé avec les tableaux. Elle permet de parcourir les éléments d'un tableau:

```
foreach ($monTableau $key => $value) {
    # instructions...
}
```

La variable `$key` permet d'accéder à la clé de l'élément actuel. Et la variable `$value` va retourner la valeur.

Même s'il est possible de parcourir un tableau avec les autres boucles, il est préférable pour des raisons de performance, d'utiliser la boucle foreach.

Include

Nous allons utiliser la fonction include() en php pour inclure un fichier externe dans notre code PHP.

Cette méthode permet d'organiser le code de notre site et d'éviter les répétitions.
Pour inclure un fichier nous allons utiliser la syntaxe:
`include('chemin/vers/le/fichier.php');` ou
`include_once('chemin/vers/le/fichier.php');`

La différence entre ces deux fonctions est que la fonction include_once va permettre d'éviter d'inclure un code déjà présent alors que la fonction include() ne fait pas cette vérification.

Liens utiles

- https://www.php.net/manual/fr/index.php
- https://openclassrooms.com/fr/courses/918836-concevez-votre-site-web-avec-php-et-mysql
- https://www.grafikart.fr/tutoriels/php
- https://pub.phyks.me/sdz/sdz/rogrammez-en-oriente-objet-en-php.html