



Cours de PHP

1) Introduction

Qu'est-ce que PHP ?

PHP (PHP : Hypertext Preprocessor), est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur http mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet. (Wikipédia)

C'est un langage qui va être exécuter coté serveur et il va permettre de générer dynamiquement de l'HTML en fonction de données provenant de la base de données ou autre.

C'est un langage qui a été créé à la base spécifiquement pour générer des pages web dynamiques, c'est pour cela qu'au jour d'aujourd'hui c'est un langage idéal pour ça.

Il fonctionne au travers d'un serveur http, ça veut dire qu'on va faire une requête http, demander par exemple une page particulière, automatiquement le serveur par exemple apache va l'interpréter et dire cette page là je n'ai pas de fichier html qui correspond mais je vais appeler PHP qui va générer le code html.

Comme je le disais aussi, PHP peut être utilisé et fonctionné comme n'importe quel langage interprété de façon locale.

C'est un langage interprété c'est-à-dire que pour fonctionner, il aura besoin d'un interpréteur. On en parlera après l'installation. Ce n'est donc pas un langage compilé, qu'on compile qu'une fois et ensuite qui peut être exécuter sans forcément une étape supplémentaire.

On dit aussi que c'est un langage impératif, ça veut dire qu'il est exécuté de manière séquentielle, donc une première ligne est exécutée ensuite on passe à la suivante, etc. On ne peut pas avoir plusieurs opérations qui peuvent être exécuter en même temps.

Il est orienté objet donc peut utiliser le système de classe, d'héritage etc., on y reviendra par après.

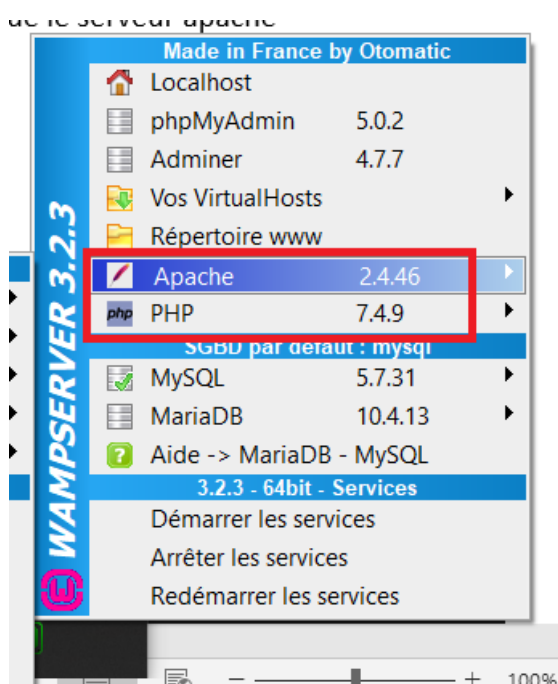
PHP a un typage qui est dit dynamique et faible, ça veut dire que contrairement à d'autres langages de programmation, il ne sera pas nécessaire lorsque l'on créera une variable de déclarer le type en amont et de dire par exemple dans cette variable qu'on aura un entier, ou une chaîne de caractère etc. Automatiquement php déterminera le type en fonction du contexte donc si on envoie un entier à une variable, il se dit c'est bon c'est un entier. Et de la même manière les variables peuvent changer de type au fur et à mesure de l'exécution du script.

C'est donc très bien pour débiter, car on ne se prendra pas trop la tête à la déclaration, mais l'inconvénient c'est que si vous ne faites pas très attention, ça pourra créer des erreurs dans vos codes.

2) Installation

Pour télécharger PHP vous avez deux solutions. La première est d'aller sur le site officiel et de télécharger puis installer PHP. Et le configurer comme vous le souhaitez.

La deuxième et celle que vous avez déjà fait c'est de télécharger WampServer et de l'installer. Wamp nous fournit directement PHP ainsi que le serveur apache.



Il faut aller ajouter la variable d'environnement pour php.

Pour vérifier quelle version vous utilisez de PHP, il y a aussi via ligne de commande en faisant `php -v`.

```
C:\Users>php -v
PHP 7.4.9 (cli) (built: Aug  4 2020 11:52:41) ( ZTS Visual C++ 2017 x64 )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies

C:\Users>
```

Voilà vous avez PHP d'installer.

Maintenant il nous faudra un **éditeur**. Il faudra télécharger et installer **Visual Studio Code**. Normalement vous l'avez déjà.

On va **créer** un **dossier** dans **C:\wamp64\www** qu'on appellera « **coursphp** ». Une fois fait, on va le lancer avec Visual studio code. Ouvrez votre ligne de commande Windows et tapez « `code .` » quand vous êtes dans le dossier **C:\wamp64\www\coursphp**.

```
C:\wamp64\www\coursphp>code .
```

Cette commande va ouvrir VS Code directement dans votre dossier.

Dans VS Code, on peut tester de **créer** un **fichier** qu'on appellera « **testos.php** » et ensuite dans ce fichier y mettre une phrase « Bonjour monsieur test ». **Attention** n'oubliez pas de sauvegarder à chaque fois que vous modifiez vos fichiers en faisant « **CTRL+S** ».

Vous pourrez ensuite ouvrir un nouveau terminal et taper :

```
PROBLÈMES  SORTIE  TERMINAL  CONSOLE DE DÉBOGAGE
PS C:\wamp64\www\coursphp> php textos.php
Bonjour monsieur test
PS C:\wamp64\www\coursphp>
```

Cette commande permet d'exécuter un fichier avec l'extension `.php`. Ici il nous affiche tout simplement le contenu. Si on avait des balises PHP il aurait exécuté le code dedans.

Pour ouvrir du code php, on utilise les balises php de cette manière

```
<?php

?>
```

C'est dans ces balises qu'on mettra le code en php. Tout ce qui sera en dehors de la balise php, sera juste afficher comme on a pu le voir pour notre première phrase.

Pour savoir ce qu'il faut mettre dedans on peut aller sur le site officiel

<https://www.php.net/manual> là se trouve toute la documentation sur le langage php.

3) Les Variables

A quoi ressemble PHP avec du HTML ?

```
<html>
<body>
<font size="2" face="Arial">Le texte en HTML</font>

<?php
// le code PHP -----
$heure = date("H\hi");
print("<font size='2\" face='\"Arial\"> et celui en PHP.</font>");
?>
<!-- retour au code HTML -->
<br><font size="2" face="Arial">Il est <?php echo $heure; ?>.</font>
</body>
</html>
```

Voici un fichier .php avec du HTML et du code PHP. Je ne vais pas vous expliquer ce que fais le code HTML, vous l'aurez compris. Intéressons-nous plutôt à la partie PHP.

On le commence en mettant la balise « < ?php... » pour dire que le code qui suit sera en PHP, jusqu'à ce qu'on la referme « ?> ».

Dedans on peut voir tout d'abord la manière d'écrire une ligne de **commentaire** en **PHP** « // » et d'ailleurs qui est la même dans beaucoup d'autres langages.

Ensuite on a une variable « \$heure », qu'on a déclaré. Qui va recevoir une date dans le format heure minute « date("H\hi") », cette fonction donne l'heure actuelle et va être placé dans la variable \$heure.

On a ensuite la fonction « print() » qui va imprimer ce qu'il y a dedans. Ici on a du code html qu'il va savoir comprendre. Il va imprimer « et celui en PHP » dans la même écriture qu'on avait au début pour html (arial taille 2).

Vous remarquerez qu'à la fin de chaque ligne de code en php, on doit rajouter un « ; ». A la fin on ferme la balise php « ?> ».

On retourne sur du HTML et ensuite on réouvre une balise PHP avec comme commande « echo \$heure ; » le mot clé echo c'est pour afficher, donc php dit, affiche-moi ce qu'il y a dans la variable \$heure.

Le résultat sera :



Le texte en HTML et celui en PHP.
Il est 09h49.

On va pouvoir tester ce code dans un fichier php.

Copier/coller le code php de la page 4 dans votre fichier « testos.php ». Ouvrir le terminal et taper « php testos.php » .

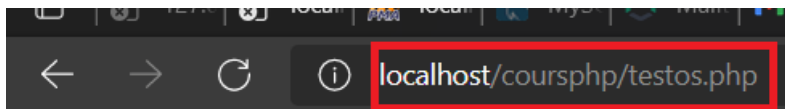
```
PS C:\wamp64\www\coursphp> php testos.php
<html>
<body>
<font size="2" face="Arial">Le blabla HTML</font>

<font size="2" face="Arial"> et celui en PHP.</font><!-- retour au code HTML -->
<br><font size="2" face="Arial">Il est 12h34.</font>
</body>
</html>
```

Vous vous rappelez qu'on a dit qu'il n'exécutera que le code php. Tout ce qui est html, il affichera juste le code. On remarquera que les balises php ne sont plus là, ni la déclaration de la variable « \$heure » ainsi que son assignation. Pareil le mot clé « print() » n'est plus, mais ça imprimer tout ce qu'il y avait dans la parenthèse. Le « echo \$heure » s'est exécuté et on peut voir qu'il nous donnera l'heure actuelle GMT +0.

Ce n'est pas très lisible tout ça.

Pour pouvoir en tirer une page web, il faudra le lancer avec localhost. Assurez-vous que Wamp est bien lancé.



Le blabla HTML et celui en PHP.
Il est 12h28.

Les **variables** en **PHP** comme en algorithmies vont **nous permettre de sauvegarder des valeurs temporairement en mémoire**, pour pouvoir ensuite les réutiliser plus tard.

Ça s'écrit de la manière suivante, \$ suivi du nom de la variable :

- \$maVariable;

Il n'est pas nécessaire comme dans d'autres langages de déclarer quoi que ce soit. On va ensuite lui assigner une valeur.

- \$maVariable = uneValeur ;

Les types basiques sont :

- Les entiers (on écrira simplement un chiffre)
- Les nombres réels (avec un point pour séparer)
- Les chaînes de caractères (guillemet simple ou double)
- Les booléens (true ou false)

Créez un autre fichier que vous appellerez **test.php** et on va créer une variable « **firstname** » en lui donnant comme valeur « **Jack** ».

```
test.php > ...
1  <?php
2      $firstname = "Jack";
3  ?>
```

Il ne faut pas oublier de mettre le « ; » qui indique à php que c'est la fin d'une instruction.

Ici pour l'instant si on le lance via localhost/coursphp/test.php dans votre navigateur, il ne va rien se passer, ça va afficher une page blanche.

Comme on l'a fait en haut pour **afficher** quelques choses en **PHP**, on a utilisé la commande « **echo** ».

```
<?php
    $firstname = "Jack";
    echo $firstname;
?>
```

On verra bien qu'il a affiché, le prénom Jack.

```
localhost/coursphp/test.php
```

Jack

En php pour afficher plusieurs chaîne de caractère l'une à la suite de l'autre (les **concaténer**), il faudra mettre un « . » entre chaque chaîne, pour concaténer celle-ci.

```
echo 'Bonjour ' . $firstname . ', bienvenue dans le monde de PHP';
```

(Exos)

- 1) Ajoutez au fichier un nom, un poids et un âge. Et affichez-les. (Montrez-moi)
- 2) Essayez d'afficher cette phrase : « Voici (prénom nom), il a (âge) ans et un poids de (poids) Kilo. ».

```
localhost/coursphp/test.php
```

Voici Jack Bauer, il a 24 ans et un poids de 75 Kilo.

- 3) Mettez en commentaire l'affichage précédent. Ajoutez au fichier 3 entiers et affichez-moi la moyenne.
- 4) Essayez de m'afficher cette phrase : « Voici 3 nombres : (nb1, nb2 et nb3), la moyenne de ces trois nombres donne : (moyenne) ».

```
localhost/coursphp/test.php
```

Voici 3 nombres : 15, 17 et 8, la moyenne de ces trois nombres donne : 13.333333333333

Je vais maintenant vous parler de la différence qu'il y a entre le guillemet simple et le guillemet double. Si vous allez dans la documentation sur le site officiel et que vous allez ensuite dans types puis dans chaînes de caractères. Vous verrez comment par exemple passer à la ligne ou encore comment rajouter dans votre affichage un guillemet double etc.

Toutes ces commandes spéciales fonctionnent avec le guillemet double. Les guillemets simples ne vont jamais interpréter correctement ces caractères spéciaux qui y sont à l'intérieur.

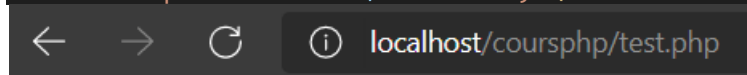
Vous pouvez même écrire vos variables dans une chaîne de caractère à l'intérieur des guillemets double, PHP va interpréter ces variables grâce aux dollars devant (\$).

```
echo " Le prénom c'est $firstname, le nom c'est $lastname";
```

Vous verrez que ça fonctionne, il affichera bien cette phrase et donnera la valeur qu'il y aura dans le prénom et le nom. Ça ne fonctionne pas avec les guillemets simples.

Pour le saut à la ligne c'est en ajoutant « \n ». Vous pourrez le rajouter aussi directement dans votre chaîne de caractère :

```
echo " Le prénom c'est $firstname, \n le nom c'est $lastname";
```




Le prénom c'est Jack, le nom c'est Bauer

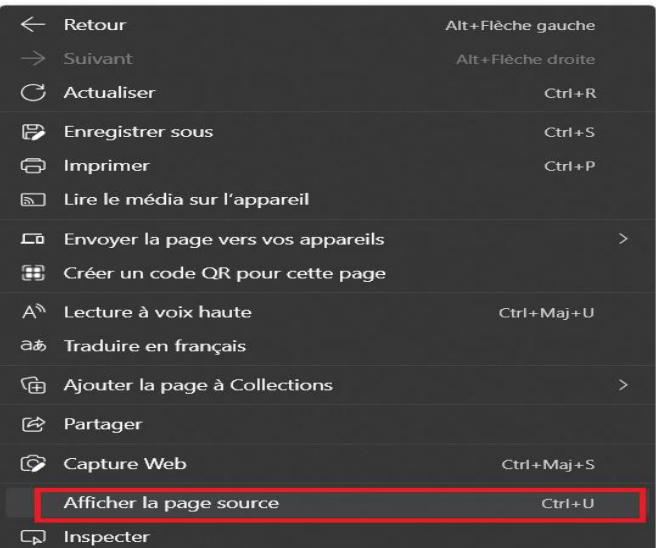
Pourquoi n'a-t-il pas fait le saut à la ligne alors qu'il était censé le faire. Par contre si je lance la commande via le terminal cela va fonctionner :

```
PS C:\wamp64\www\coursphp> php .\test.php
Le prénom c'est Jack,
le nom c'est Bauer
PS C:\wamp64\www\coursphp>
```

D'ailleurs si je vais dans mon navigateur et que je clic droit pour afficher le code source

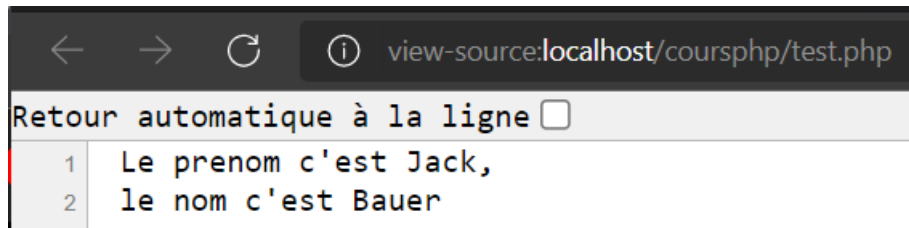


Le prénom c'est Jack, le nom c'est Bauer



- Retour (Alt+Flèche gauche)
- Suivant (Alt+Flèche droite)
- Actualiser (Ctrl+R)
- Enregistrer sous (Ctrl+S)
- Imprimer (Ctrl+P)
- Lire le média sur l'appareil
- Envoyer la page vers vos appareils >
- Créer un code QR pour cette page
- Lecture à voix haute (Ctrl+Maj+U)
- Traduire en français
- Ajouter la page à Collections >
- Partager
- Capture Web (Ctrl+Maj+S)
- Afficher la page source (Ctrl+U)**
- Inspecter

Quand on va dans le page source on a cela comme résultat :



On voit donc bien qu'il a fait le retour à la ligne.

Il n'affiche pas dans la page web, parce qu'en HTML le saut de ligne ne se fait pas avec `\n` mais bien avec `
`. Pour faire en sorte que dans notre page web, il y a le saut de ligne modifier l'écho en remplaçant le `\n` par le `
` :

```
echo " Le prénom c'est $firstname, <br> le nom c'est $lastname";
```

A noter que par convention la plupart des programmeurs utilisent les guillemets simples pour la lisibilité.

```
echo 'Le prénom c\'est ' . $firstname . ', le nom c\'est ' . $lastname;
```

Après les chaînes de caractères et les nombres, nous avons les **booléens**. Les variables sont soit vraie, soit fausse. Nous y reviendrons dans un instant quand je parlerai des conditions. En fonction que la condition soit vraie ou fausse, il exécutera différentes choses.

```
$estValide = true;  
echo $estValide;
```

Si vous tapez ça, vous verrez qu'il affichera « 1 » car le TRUE en informatique vaut 1 et le FALSE vaut 0.

Pour finir nous avons aussi les variables NULL, c'est quand la variable on ne met rien dedans, elle est vide.

```
$adresse = null;  
echo $adresse;
```

Ça n'affichera rien.