Chapitre 3
Le langage PHP

1. Comment écrire du PHP ?

Retournez dans votre éditeur Visual Studio Code et cliquez sur l'onglet **index.php**.

Nous allons écrire notre premier code dans ce fichier.

Tous les noms de fichiers PHP doivent se terminer par **.php** (exemple : monFichier.php). Évitez les caractères spéciaux dans vos noms de fichiers (/, >, <, ?, !...) et les caractères accentués (é, à, ù…) ainsi que les espaces.

Si vous donnez à votre fichier un nom autre que **index.php**, vous devrez le préciser sur le navigateur dans la barre d'adresse (nous verrons cela plus tard).

Nous allons commencer par écrire du code dans notre fichier **index.php**. Tout code PHP est encastré dans les mêmes instructions :

```
<?php
... quelque chose en php
?>
```

Bien que votre fichier se termine par **.php**, il faut en effet lui préciser ces balises à l'intérieur pour écrire du PHP. Pourquoi ?

Parce que votre fichier PHP peut aussi contenir du texte libre ou du code HTML qui sera interprété tel quel par le navigateur.

Exemple

```
<h1>Mon titre en HTML</h1>
<?php
// mon code PHP
?>
```

La page affichée commencera ici par un titre en gras : Mon Titre en HTML.

Ainsi, un fichier PHP (avec l'extension .php) peut contenir du code mixte avec des parties en HTML et des parties en PHP.

```
Ici du code HTML
<?php
Ici du code PHP
?>
ici du code HTML
<?php
ici du code PHP
 ?>
```

Il peut aussi, ce qui sera notre cas avec Symfony, ne contenir que du code PHP :

```
<?php
Ici du code PHP
?>
```

Lorsqu'un fichier ne contient que du code PHP, on peut s'affranchir (et c'est mieux pour éviter l'affichage de tout caractère qui traînerait à la suite dans la page) de mettre la balise de fin ?>.

Exemple

```
<?php
Ici du code PHP
```

Les commentaires (lignes qui ne seront pas interprétées par le serveur) sont précisés par // ou pour un bloc de lignes /* … */

Il est conseillé de mettre des commentaires dans votre code (pas trop), pour illustrer ce que fait celui-ci et pour mieux s'y retrouver lorsque vous ou quelqu'un d'autre reprendrez le développement du code.

Exemple

```
<?php
// commentaire sur une ligne
*/Exemple de commentaire
sur
plusieurs lignes */
?>
```

Visual Studio Code vous permet d'insérer automatiquement des commentaires en sélectionnant la ou les lignes désirées puis [Ctrl] / (mêmes touches pour désélectionner).

2. Les bases du langage : votre premier Hello World !

Nous allons juste écrire une ligne de code permettant d'afficher sur le navigateur :

```
Hello World !
```

On a l'habitude de souvent écrire du texte en anglais, la langue la plus utilisée en informatique, mais vous pouvez toujours écrire « Bonjour le monde ! » si vous le souhaitez.

L'instruction PHP qui permet d'afficher quelque chose sur le navigateur s'appelle : `echo`.

▶Écrivez ce code dans le fichier index.php :

```
<?php
echo 'Hello World !';
?>
```

On remarque qu'il faut mettre le texte entre les ' (quotes) pour le distinguer du code, et que l'instruction se termine par un point-virgule.

Remarque

Toutes les instructions PHP se terminent par un point-virgule. Si vous l'oubliez, vous aurez une erreur sur votre navigateur.

▶Attention, n'oubliez pas de sauvegarder votre fichier : [Ctrl] **S** (il faudra à chaque fois que vous modifiez un fichier, sauvegarder les modifications).

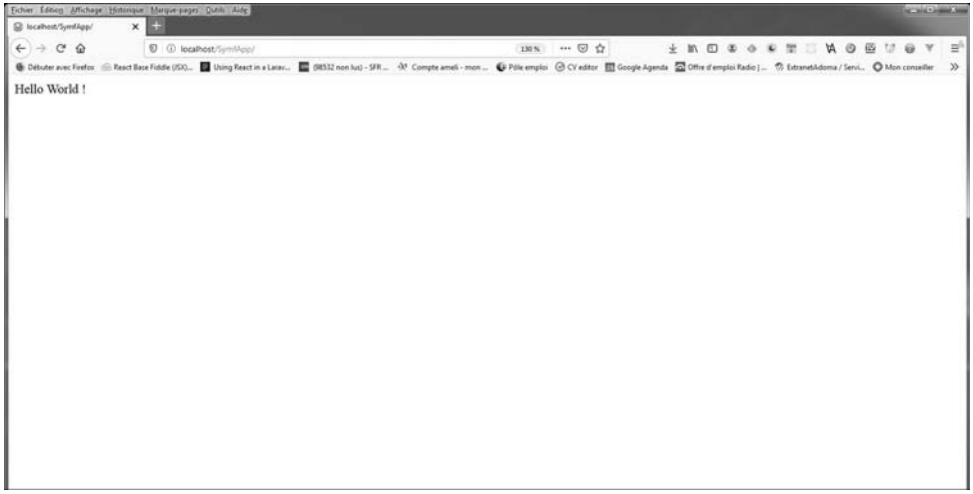
Comment voir le résultat de votre code sur votre navigateur comme le verrait un internaute ?

Si votre WAMP est toujours allumé (**W** vert dans la barre des tâches, sinon pensez à le redémarrer). Ouvrez un navigateur (le mieux est d'ouvrir Firefox) et tapez l'adresse suivante dans la barre d'adresse : localhost/SymfApp.

Remarque

localhost est le nom du serveur installé sur votre machine par WAMP.

Le fichier index.php s'ouvrira tout seul car c'est celui qui est ouvert par défaut, et vous verrez votre texte s'afficher :



Voilà, vous venez d'écrire votre premier code PHP.

- Revenez sur votre fichier index.php dans Visual Studio Code (qui sera nommé Vscode dans la suite) et modifiez le titre :

```
<?php
echo 'Hello everybody';
?>
```

- Sauvegardez (touches [Ctrl] **S**) et rafraîchissez votre navigateur. Vous verrez votre modification effectuée sur le navigateur.

Nous allons apprendre quelques instructions de PHP.

Il ne s'agit pas ici d'explorer le langage PHP dans ses moindres détails, mais d'avoir les outils nécessaires pour être à l'aise avec le framework Symfony.

Les outils que nous allons voir sont vraiment indispensables. Alors plongez-vous dans cette initiation, minutieusement. Entraînez-vous en faisant vos propres exemples dans le fichier **index.php**.

3. Les variables en PHP

Les premiers de ces outils sont **les variables**.

Une variable permet de stocker une information pour pouvoir la réutiliser autant de fois qu'on le souhaite dans le code. Nous pouvons également faire des opérations entre des variables (les ajouter, les multiplier…).

Toutes les variables en PHP commencent par un $. C'est comme cela qu'on les identifie. Le nom d'une variable doit respecter les mêmes règles que les noms de fichiers (pas de caractères spéciaux, de caractères accentués, ni d'espace).

Exemple de variable : `$texte`

Si vous voulez utiliser des noms de variables avec plusieurs mots, l'usage est de distinguer chaque mot sauf le premier par une majuscule (ce n'est pas obligatoire, mais c'est un standard d'écriture). C'est ce qu'on appelle la norme camelCase ou dromedaryCase.

Exemple : `$monTexte`

Vous pouvez créer dans votre code autant de variables que vous le souhaitez.

Toutes les variables sont supprimées à la fin de l'exécution du code.

Il est possible d'affecter du texte à une variable :

```
<?php
$monTexte='Hello World !';
?>
```

À quoi cela sert-il ?

À chaque fois que vous voudrez afficher ce texte n'importe où dans votre code, vous utiliserez le nom de la variable. L'un des intérêts est que si vous voulez changer le texte partout où vous l'avez inséré (imaginez que vous l'ayez inséré 100 fois !), il vous suffit de changer le texte dans votre variable et il se changera automatiquement partout.

Pour afficher le contenu de votre variable, vous utiliserez toujours l'instruction `echo`, mais cette fois sans les quotes :

```
<?php

$monTexte='Hello World !';

echo $monTexte;

?>
```

Ce code donne le même résultat que précédemment (vous pouvez tester, sans oublier de faire [Ctrl] **S** à chaque fois que vous modifiez votre code).

Il est également possible d'insérer une variable dans un texte à afficher. On utilise pour cela les **"** (double quotes) :

```
<?php

$monNom='Yves';

echo "Hello $monNom !";

?>
```

Ce code affichera `Hello Yves !` dans le navigateur.

Une autre façon de faire est d'utiliser la concaténation pour juxtaposer plusieurs chaînes de caractères. On utilise l'opérateur **.** (point).

Voici ce que ça donne :

```
<?php

$monNom='Yves';

echo "Hello ".$monNom." !";

?>
```

Ici, la chaîne de caractères `"Hello"` est concaténée à la variable `$monNom` qui est elle-même concaténée à la chaîne de caractères `" !"`.

On appelle aussi les chaînes de caractères, des **strings**. C'est ce mot qui sera utilisé par la suite dans l'ouvrage.

Il est possible également d'insérer des nombres dans une variable (pour faire des opérations par exemple). Pour distinguer les nombres des strings, on omet les quotes.

Exemple : `$prix=20; $tva=0.196;`

Voici par exemple comment afficher une quantité de produits et une TVA avec des variables :

```
<?php

$prix=20;
$quantite=10;
$tva=0.196;

$total=$prix*$quantite*$tva;

echo "Total Prix TTC : $total euros";

?>
```

Remarque

** est l'opérateur de multiplication (les opérateurs arithmétiques sont : +, -, *, /).*

Vous pouvez tester.

Le résultat donnera : `Total Prix TTC : 39.6 euros`.

Les variables sont très utiles, car elles permettent d'agréger plusieurs données dans tout le code.

4. Les déclarations de type en PHP

Vous l'avez compris, les variables sont définies suivant leur type. Nous appelons `TYPE DE VARIABLE`, la nature de la valeur qu'elle peut stocker. Est-ce que la variable est de type numérique (exemple : *$prix=20;*) ou alphanumérique (exemple : *$nom='yves';*) ?

Selon le type de la variable, son comportement et son affichage diffèrent. Par exemple, je peux faire des opérations sur les variables de type numérique, mais pas sur les variables de type alphanumériques.

Il existe en PHP, différents types de variables, les voici énumérées :

Null : la variable ne peut prendre qu'une seule valeur, la valeur null. Elle existe, mais ne contient rien. On peut déclarer des variables sans leur assigner de valeur. Il suffit d'écrire : *$mavariable=null ;*. Cette variable sera censée contenir une valeur plus tard, dans le code.

Bool : ces variables ne peuvent prendre que deux valeurs : *true* ou *false*. Nous verrons dans la section suivante (Les structures de contrôle) l'intérêt d'un tel type.

Int : ce sont des variables numériques entières. Elles ne peuvent contenir que des nombres entiers (pas de nombres décimaux). Exemple : *$prix=10 ;*

Float : ce sont des variables numériques à virgule flottante (autrement dit, des nombres décimaux aussi connus comme "floats", "doubles", ou "nombres réels"). Ces variables peuvent être spécifiées en utilisant les syntaxes suivantes :

$a = 1.234;

$b = 1.2e3;

$c = 7E-3; cette dernière notation est la notation scientifique (7E-3 signifie 0.007).