

OPERATIONS SUR LES FICHIERS EN PHP

En PHP, nous pourrions avoir recours aux fichiers pour effectuer différentes tâches, notamment lorsque l'on veut stocker définitivement une information sur notre serveur.

En effet, les variables, bien que plus simples à prendre en main, ne permettent de stocker que temporairement une information.

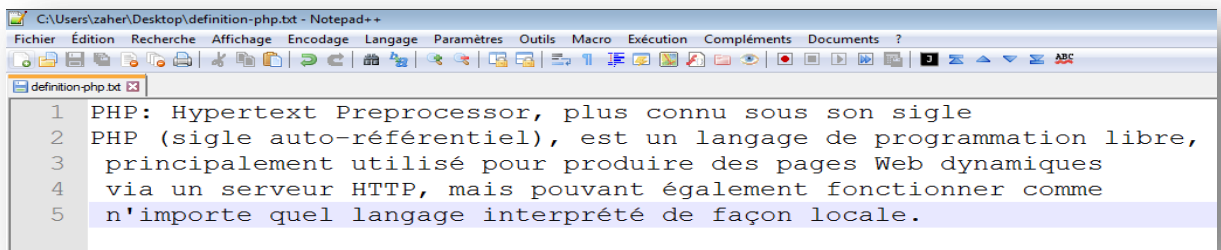
PHP possède plusieurs fonctions permettant de créer, de lire, d'éditer ou de fermer des fichiers. Nous allons passer en revue ces différentes fonctions dans ce chapitre.

Toutefois notez que dans la plupart des cas il est plus pertinent d'utiliser les bases de données que les fichiers.

Ouvrir ou créer un fichier

Nous allons déjà commencer par créer un fichier texte (avec extension .txt donc). Vous pouvez mettre le texte que vous voulez dans ce fichier.

Pour ma part, je vais me contenter de copier-coller une définition du PHP issue du site Wikipedia. J'appellerai ce fichier « definition-php.txt » :



The screenshot shows a Notepad++ window with the file 'C:\Users\zaher\Desktop\definition-php.txt'. The text inside is a definition of PHP: '1 PHP: Hypertext Preprocessor, plus connu sous son sigle', '2 PHP (sigle auto-référentiel), est un langage de programmation libre,', '3 principalement utilisé pour produire des pages Web dynamiques', '4 via un serveur HTTP, mais pouvant également fonctionner comme', '5 n'importe quel langage interprété de façon locale.'

Pour ouvrir notre fichier à l'intérieur d'une page PHP, nous allons utiliser la fonction `fopen()`, abréviation de « file open ».

Cette fonction a besoin de deux arguments pour fonctionner : le premier contient le nom du fichier à ouvrir et le deuxième correspond au mode d'ouverture de notre fichier comme ceci :



The screenshot shows an HTML document with a PHP block. The PHP code is: '\$definition = fopen("definition-php.txt", "r+");'. The HTML structure includes <!DOCTYPE html>, <html>, <head> with <title>Les fichiers en PHP</title> and <meta charset = "utf-8"/>, </head>, <body>, <?php, the PHP code line, <?>, </body>, and </html>.

Lorsque l'on ouvre un fichier avec `fopen()`, la fonction va nous renvoyer une information que nous allons toujours stocker dans une variable. Cela nous servira à effectuer différentes opérations sur notre fichier par la suite, comme fermer le fichier.

Attention : ici, nous n'avons fait qu'ouvrir le fichier dans notre code PHP, nous n'avons pas encore demandé à le lire !

Concernant les modes d'ouverture d'un fichier, il en existe huit différents :

MODE	DESCRIPTION
r	Ouvre un fichier en lecture seule. Pas de modification possible.
r+	Ouvre un fichier en lecture et en écriture.
a	Ouvre un fichier en écriture seule. Si le fichier n'existe pas, en crée un nouveau.
a+	Ouvre un fichier en lecture et en écriture. Si le fichier n'existe pas, en crée un nouveau.
w	Ouvre un fichier en écriture seule. Si le fichier existe déjà, supprime le contenu préexistant. Si il n'existe pas, en crée un nouveau.
w+	Ouvre un fichier en lecture et en écriture. Si le fichier existe déjà, supprime le contenu préexistant. Si il n'existe pas, en crée un nouveau.
x	Crée un nouveau fichier ouvert en écriture seulement. Retourne la valeur FALSE si le fichier existe déjà.
x+	Crée un nouveau fichier ouvert en lecture et en écriture. Retourne la valeur FALSE si le fichier existe déjà.

Fermer un fichier

Pour fermer un fichier, nous allons cette fois utiliser la fonction `fclose()`. Cette fonction va prendre comme unique argument la variable dans laquelle on a stocké les informations renvoyées par la fonction `fopen()`.

Cette fonction s'utilise très simplement, comme ceci :

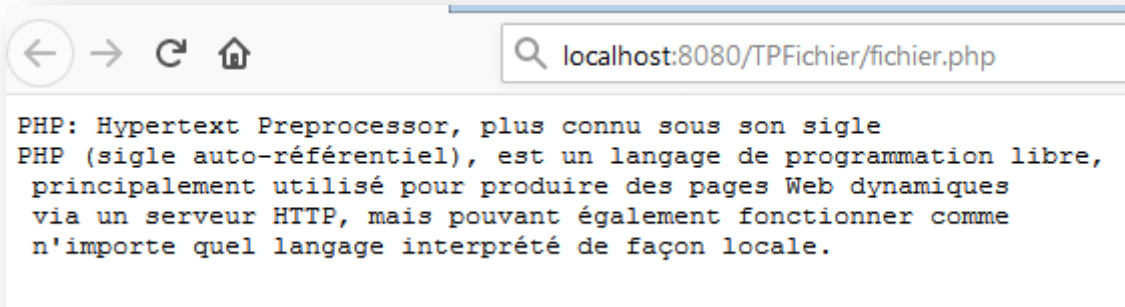
```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Les fichiers en PHP</title>
5      <meta charset = "utf-8"/>
6  </head>
7  <body>
8      <?php
9          $definition = fopen("definition-php.txt", "r+");
10         fclose($definition);
11     ?>
12 </body>
13 </html>
```

Lire un fichier

Pour lire un fichier, la méthode la plus simple est d'utiliser la fonction `fread()`. Cette fonction prend deux paramètres : le premier correspond au fichier à lire, et le second spécifie le maximum de bytes qui doivent être lus.

On peut ensuite tout simplement utiliser l'instruction `echo` pour retourner le résultat de `fread()`. On peut également stocker le résultat dans une variable et `echo` cette variable si l'on souhaite pouvoir s'en resservir par la suite.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Les fichiers en PHP</title>
5     <meta charset = "utf-8"/>
6   </head>
7   <body>
8     <?php
9       $definition = fopen("definition-php.txt", "r+");
10      $affichage = fread($definition, 1000);
11      echo $affichage;
12      fclose($definition);
13    ?>
14  </body>
15 </html>
```

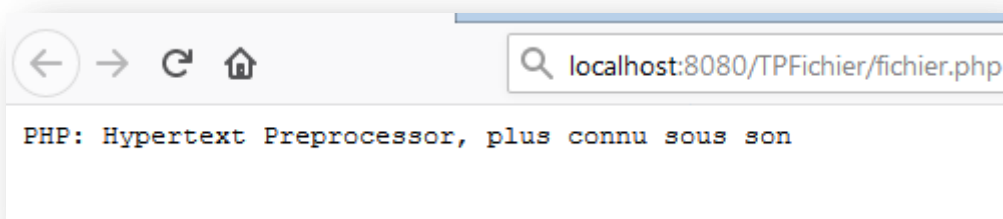


On peut également utiliser la fonction `fgets()` pour lire dans un fichier. Cette fonction va lire un fichier ligne par ligne, donc si votre fichier fait plusieurs lignes et que vous désirez le lire entièrement, vous devrez utiliser une boucle ou une condition.

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Les fichiers en PHP</title>
5          <meta charset = "utf-8"/>
6      </head>
7      <body>
8          <?php
9              $definition = fopen("definition-php.txt", "r+");
10             $affichage = fgets($definition, 1000);
11             echo $affichage;
12             fclose($definition);
13         ?>
14     </body>
15 </html>

```



On peut lire un fichier ligne par ligne jusqu'au bout assez simplement en utilisant une condition avec la fonction `feof()` qui vérifie lorsque la fin du fichier (End Of File) est atteinte. On s'y prendrait comme cela :

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Les fichiers en PHP</title>
5          <meta charset = "utf-8"/>
6      </head>
7      <body>
8          <?php
9              $definition = fopen("definition-php.txt", "r+");
10             while (!feof($definition)){
11                 echo fgets($definition);
12             }
13             fclose($definition);
14         ?>
15     </body>
16 </html>

```

Que nous dit cette condition `while` ? On utilise un point d'exclamation au début qui signifie la négation.

Cette condition va donc echo le fichier `definition-php.txt` ligne par ligne tant que la fin du fichier n'est pas atteinte. Vous pouvez la tester, ça fonctionne !

Finalement, il existe une troisième fonction nous permettant de lire un fichier : la fonction `fgetc()`. Cette fonction va lire un fichier caractère par caractère. En pratique, on ne l'utilisera pas souvent car elle n'a pas de réel avantage par rapport aux deux autres présentées ci-dessus.

La place du curseur en PHP

Les plus curieux d'entre vous devraient déjà s'être posés la question : où est placé notre curseur lorsqu'on ouvre ou écrit dans un fichier ?

Le curseur PHP correspond à l'endroit où va être exécuté votre prochaine commande en PHP.

Voici un rapide résumé d'où se trouve le curseur après chaque commande utilisée en PHP, afin que vous sachiez où vous en êtes :

MODE / FONCTION	POSITION DU CURSEUR
<code>r / r+</code>	Pointe au début du fichier
<code>a / a+</code>	Pointe à la fin du fichier
<code>w / w+</code>	Pointe au début du fichier
<code>fgets()</code>	Pointe à la fin de la ligne lue
<code>fgetc()</code>	Le curseur se place au niveau du caractère suivant

Pour changer la position du curseur, vous devez utiliser la fonction `fseek()`. Cette fonction prend deux arguments : le fichier pour lequel on doit modifier la position du curseur, et la nouvelle position voulue en bytes.

Notez que si vous avez ouvert votre fichier avec le mode `a` ou `a+`, les données ajoutées par la suite seront toujours ajoutées à la fin du fichier, même si vous utilisez `fseek()`.

Écrire dans un fichier

Pour écrire dans un fichier, vous devez au préalable vous assurer d'avoir les permissions nécessaires.

Pour écrire dans un fichier, nous allons utiliser la fonction `fwrite()`. Cette fonction utilise deux arguments : le premier correspond au fichier dans lequel écrire tandis que le second va être le texte à écrire.

Cette fonction s'utilise de la manière suivante :


```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Les fichiers en PHP</title>
5     <meta charset = "utf-8"/>
6   </head>
7   <body>
8     <?php
9       $definition = fopen("definition-php.txt", "a+");
10      fwrite($definition, "Cette définition me semble très technique, non ?");
11      fclose($definition);
12    <?>
13  </body>
14 </html>

```

Et

voilà le résultat après un rafraichissement de notre page definition-php.txt :

```

1 PHP: Hypertext Preprocessor, plus connu sous son sigle
2 PHP (sigle auto-référentiel), est un langage de programmation libre,
3 principalement utilisé pour produire des pages Web dynamiques
4 via un serveur HTTP, mais pouvant également fonctionner comme
5 n'importe quel langage interprété de façon locale.
6 Cette définition me semble très technique, non?

```

Attention : notez bien que lorsque vous écrivez au début d'un fichier ou au début d'une ligne, votre nouveau texte va venir écraser l'ancien présent à cette place. Vous n'écrirez pas devant le texte, mais à la place de l'ancien texte.

Exercice :

1. Faites une recherche sur fputs ,fseek, base_convert
2. TP :

Ecrire un script php qui permet de lire des nombres décimaux à partir d'un fichier texte nommé "decimal.txt", et stocke leurs équivalents en binaire dans un autre fichier "binaire.txt".

decimal.txt	binaire.txt
12	1100
21	10101
4	100
17	10001

Convertisseur