

Escritura en archivos con PHP

Siguiendo con la gestión de archivos por medio de PHP, en este artículo veremos los pasos elementales para la creación y escritura de un archivo texto por medio de esta tecnología de lado servidor. Tras haber visto como funciona la [lectura secuencial de un archivo](#), podemos imaginar que escribir sobre éste no debe de resultar mucho más complicado.

Por otra parte, las posibilidades que estas dos operaciones nos pueden ofrecer conjuntamente son realmente sorprendentes. Sin ir más lejos, y guardando las distancias, escribir y leer archivos puede en cierta manera sustituir muy primariamente a una base de datos. En efecto, si por diversas razones (hosting, presupuesto, conocimientos...) nos resulta difícil la puesta en marcha de una base de datos elemental, siempre podremos solventar el inconveniente almacenando nuestros datos en archivos que luego podrán ser leídos. Por supuesto, este método no tiene nada de seguro ni de versátil y sólo es valido para un sitio sin información confidencial y con poca cantidad de datos.

Podemos pensar también en crear documentos dinámicos a partir de datos introducidos en un formulario: cartas, páginas HTML y otros.

Otro ejemplo particularmente práctico es la creación dinámica de archivos que nos ahorren recursos de servidor. Imaginemos que tenemos una página, o archivo, en nuestro sitio que carga muy frecuentemente y que realiza constantemente llamadas a bases de datos o ejecuta scripts medianamente largos. Si el contenido que estamos mostrando es el mismo para todos los usuarios y no tiene necesidad de ser actualizado constantemente, podemos contentarnos con crear un script accesorio que ejecute una única vez el script principal y que almacene su resultado en forma de archivo HTML que será en realidad el que mostraremos a nuestros visitantes. De esta forma, evitamos por una parte la ejecución masiva de un mismo script con el consiguiente ahorro de recursos y por otra automatizamos la actualización de una determinada página o sección ejecutando periódicamente el script accesorio.

La escritura de archivos pasa, como es de esperar, por la previa [apertura de archivo en un modo apropiado](#). Una vez abierto, el paso siguiente será introducir por medio de la función *fwrite*, o su alias *fputs*, la cadena que deseamos incluir en nuestro archivo.

Para ejemplificar esta nueva función de escritura y combinarla con la de lectura, *fgets*, os proponemos este contador inspirado en una nota de la [página oficial de PHP](#):

```
<?
function incremento_contador($archivo)
{
// $archivo contiene el numero que actualizamos
$contador = 0;

//Abrimos el archivo y leemos su contenido
$fp = fopen($archivo,"r");
$contador = fgets($fp, 26);
fclose($fp);

//Incrementamos el contador
++$contador;
```

```

//Actualizamos el archivo con el nuevo valor
$fp = fopen($archivo,"w+");
fwrite($fp, $contador, 26);
fclose($fp);

echo "Este script ha sido ejecutado $contador veces";
}

?>

```

Aquí podéis ver el resultado producido cuando llamamos a esta función.

Como en otros ejemplos, el script es expresado en forma de función para que sea más sencilla su reutilización. Las etapas que llevamos a cabo son verdaderamente cortas y comprensibles:

- Iniciamos nuestra variable *contador*.
- Abrimos el archivo en modo lectura y extraemos el valor actual del contador leyendo la primera y única línea. Cerramos el archivo.
- Aumentamos de una unidad el valor de *contador*.
- Abrimos el archivo y lo sobrescribimos (modo +w) con el valor contador modificado.

Ni que decir tiene que para que este tipo de scripts funcionen, el archivo al que queremos acceder ha de estar autorizado para lectura y escritura.

La función fwrite puede ser utilizada también para enviar datos en otros tipos de aperturas como son las de sockets o de programas. Pero esto ya es otra historia... Gestión de archivos por PHP

El tratamiento de archivos resulta ser una práctica muy común en cualquier sitio web. Muy a menudo nos vemos en la necesidad de procesar un texto para cambiarle el formato, buscar una cadena en su interior o cualquier otro tipo de operación.

PHP propone un sinfín de [funciones para la gestión de archivos](#) que van desde las más elementales de apertura, lectura y cierre a otras más rebuscadas como el cálculo de espacio en el disco duro, tamaño del archivo, gestión de derechos de acceso...

En este artículo pretendemos mostraros cuáles son las funciones más esenciales para el tratamiento de archivos para posteriormente ejemplificarlas en un par de scripts que os pueden resultar útiles:

Funciones de gestión de archivos		
Función	Descripción	Sintaxis
copy	Copia un archivo	copy(\$origen,\$destino)
rename	Cambia el nombre del archivo de <i>\$antes</i> a <i>\$despues</i>	rename(\$antes,\$despues)
unlink	Borra el archivo	unlink(\$archivo)

Funciones para la lectura de archivos

Función	Descripción	Sintaxis
fopen	Abre un archivo y le asigna un identificador id. Veremos el modo más adelante	\$id = Fopen(\$archivo, \$modo)
fgets	Lee una línea de un archivo hasta un número máximo de caracteres	fgets(\$id,\$max)
fwrite	Escribe una cadena dentro del archivo	fwrite(\$id, \$cadena)
fseek	Avanza o retrocede el puntero del archivo un cierto número de posiciones	fseek(\$id,\$posiciones)
feof	Comprueba si el puntero que lee el archivo ha llegado al final	feof(\$id)
fpassthru	lee completamente el archivo y lo muestra	fpassthru(\$id)
fclose	Cierra el archivo abierto previamente	fclose(\$id)

Las operaciones más elementales, copia, borrado y cambiar el nombre, requieren únicamente el nombre (y path) del archivo sobre el cual se ejerce la operación. Para operaciones más complejas, como la lectura de líneas o la escritura de texto dentro del archivo, se requiere de una previa apertura del archivo al cual le asignaremos un identificador *\$id*.

Una vez abierto el archivo, podremos desplazarnos a lo largo de él por medio de un puntero imaginario que avanza o retrocede por las líneas de texto y mediante el cual nos situaremos en el lugar escogido para insertar, modificar o simplemente copiar una cadena.

Existen distintos modos de apertura que nos permiten definir las acciones que podemos realizar sobre el archivo. Aquí os mostramos los diferentes modos que, como veréis, son de lo más variado:

Modos de apertura de archivos	
Sintaxis	Descripción
'r'	Sólo lectura
'r+'	Lectura y escritura
'w'	Sólo escritura
'w+'	Lectura y escritura. Suprime el contenido anterior si se escribe. El archivo es creado si no existe.
'a'	Sólo escritura. El archivo es creado si no existe y el puntero se coloca al final.
'a+'	Lectura y escritura. El archivo es creado si no existe y el puntero se coloca al final.

A notar que si tratamos con archivos en binario hemos de colocar una *b* delante del modo (ej. *ba*, *bw+*, ...)

Recordamos que esta lista no es más que una recopilación y que muchas [otras funciones relacionadas](#) pueden sernos también útiles.