

Ficheros y Directorios

Antonio Espín Herranz

Contenidos

- Gestión del Sistema de Archivos.
- Subida de ficheros al Servidor.
- Gestión de directorios.

Entrada / Salida

- Cuando trabajamos con el sistema de archivos tenemos que realizar tres operaciones:
 - Abrir el fichero.
 - Procesar el contenido del fichero: leer, escribir, etc.
 - Cerrar el fichero.

Abrir fichero

- La función:
 - **fopen**(fichero, modo [, ruta_include,[contexto]]);
 - Fichero: Una cadena con la ruta del fichero. Puede ser una url o una ruta local.
 - OJO si es una url debe de estar activada la directiva: **allow_url_fopen** en php.ini.
 - Modo: La forma en que se abre el fichero: lectura, escritura. Y el tipo texto / binario, por defecto será binario.

Abrir fichero

- ruta_include: Con true, la función fopen utiliza los directorios especificados en la directiva: **include_path** de php.ini.
- Contexto: es opcional y permite que los protocolos ssl y tsl, puedan recibir el contexto de la conexión.
- fopen se combina con die por si se produce un error.
 - \$variable = fopen(nombre_fichero, modo) or die(mensaje);

Modos

- Sólo Lectura: **r**
- Lectura y escritura: **r+**
- Solo escritura. Borra el contenido anterior: **w**
- Lectura y escritura: Borra el contenido anterior: **w+**
- Solo Escritura. Conserva contenido: **a**
- Lectura y Escritura: Conserva el contenido anterior: **a+**
- Crea y abre para escritura. Si existe → error, sino false: **x**
- Crea y abre para esc / lec. Si existe → error, si no false: **x+**
- **Por defecto es binario, para texto indicar t.**

Procesar el contenido

- Para recorrer:
 - `boolean feof($fichero)` → Devuelve true si se alcanza el final del fichero.
- Para leer:
 - `string fread($fichero, int bytes_a_leer)`
- Para leer carácter a carácter:
 - `string fgetc($fichero)`
- Leer línea entera:
 - `string fgets($fichero, [int numero_bytes])`
 - Leerá `numero_bytes - 1`, puede leer menos si alcanza `\n` o fin fichero. Por defecto vale 1024.

Procesar el contenido

- `string fgetss($fichero [,int bytes [, mostrar_tags]])`
 - Igual que la anterior pero lee un fichero html y podemos quitar los tags menos los que indiquemos.
 - Ejemplo:
 - `$buffer = fgetss($fichero, 4096, '<body>
');`

Cerrar fichero

- `boolean fclose($variable).`
 - Devuelve true si no hay ningún problema.
 - Fuerza a que se grabe todo el contenido del buffer en el fichero.

Escribir en un fichero

- `int fputs($fichero, string cadena [, int longitud])`
- `int fwrite($fichero, string cadena [, int longitud])`
 - Si longitud es mayor que el número de bytes se escribe completa.

Acceso directo

- `fseek($fichero, int posicion [,int base])`
 - Devuelve 0 si se pudo posicionar, -1 en caso contrario.
 - Base:
 - `SEEK_SET`: Desde el inicio del fichero.
 - `SEEK_CUR`: A partir de la posición actual.
 - `SEEK_END`: Desde el final del fichero.
- `int rewind($fichero)`: Me coloco al principio.
- `int ftell($fichero)`: Recupera la posición del fichero.

Ficheros CSV

- Ficheros separados por comas.
- `array fgetcsv($fichero [, int longitud [, string separador [, string delimitador]]]);`
 - Longitud: El max. Tamaño a leer. Si no se indica no hay límite, pero es mas lento.
 - Separador: Por defecto la coma.
 - Delimitador: representa el delimitador de los campos cuando estos incluyen varias palabras.

Ficheros CSV

- `int fputcsv($fichero , array campos [, string separador [, string delimitador]]);`
 - campos: Array con los campos a escribir.
 - Separador: Por defecto la coma.
 - Delimitador: representa el delimitador de los campos cuando estos incluyen varias palabras.
 - Esta función devuelve el número de campos escritos.

Subida de Ficheros

- Necesitamos:
 - Formulario con el parámetro **enctype** con el valor: **“multipart/form-data”**.
 - Etiqueta input de tipo file.
 - Campo hidden para poder indicar el tamaño máximo (en bytes) del fichero a subir. Denominado: **MAX_FILE_SIZE**.
 - La directiva **upload_max_filesize** del php.ini limita el tamaño del fichero a subir.

Directivas

Directiva	Descripción
<code>file_uploads</code>	Activa o desactiva la capacidad de subir archivos (<code>On</code> para permitir).
<code>upload_max_filesize</code>	Tamaño máximo permitido por archivo (ej. <code>10M</code> para 10 megas).
<code>post_max_size</code>	Tamaño máximo total de datos enviados por POST (debe ser mayor que <code>upload_max_filesize</code>).
<code>max_file_uploads</code>	Número máximo de archivos que se pueden subir en una sola petición.
<code>upload_tmp_dir</code>	Directorio temporal donde se almacenan los archivos antes de procesarlos.

`$_FILES`

- Contiene información de los ficheros subidos.
 - Tiene como índice de filas: `'nombre_fichero'`. Será el nombre que le hayamos dado al control file de HTML.
 - Columnas: `'name'`, `'size'`, `'type'`, `'tmp_name'`, `'error'`.
- Al recibir el fichero el nombre se almacena en `tmp_name`.
- Si el tamaño del fichero supera `MAX_FILE_SIZE` la propiedad `size` toma el valor 0 y `tmp_name` será `none`.

Funciones

- **is_uploaded_file** (fichero): Devuelve true si el fichero ha sido cargado por HTTP y false en caso contrario.
 - OJO preguntar por:
`$_FILES['nombre_fichero']['tmp_name']`
- **move_uploaded_file**(origen, destino): Mueve el nombre del fichero, será algo así:
 - `move_uploaded_file(
 $_FILES['nombre_fichero']['tmp_name'],
 $_FILES['nombre_fichero']['name']);`

Códigos de error

- Se almacenan en `$_FILES['miFichero']['error']`
 - `UPLOAD_ERR_OK: 0`, OK.
 - `UPLOAD_ERR_INI_SIZE: 1`, Mayor tamaño del fichero que la directiva `upload_max_file`.
 - `UPLOAD_ERR_FORM_SIZE: 2`, Mayor tamaño del fichero que `MAX_FILE_SIZE`.
 - `UPLOAD_ERR_PARTIAL: 3`, Solo se ha subido parte.
 - `UPLOAD_ERR_NO_FILE: 4`, El usr no ha seleccionado fichero.
 - `UPLOAD_ERR_NO_TMP_DIR: 6`, No existe el directorio temporal del PHP.

Técnica Buffering

- Esta técnica utiliza buffers de salida.
- En vez de enviarlo al navegador se va almacenando en un buffer de salida.
- El contenido del buffer se puede modificar y luego ya se envía al navegador.
- Hay que activar la captura mediante la función **ob_start()**, se le puede pasar el nombre de una función para realizar algún tratamiento sobre los datos del buffer.
- La llamada a la función se realiza cuando llamamos a la función **ob_end_flush()** → envía el contenido del buffer y termina el modo captura.

Ejemplo

```
<?php
function tratar($cadena)
{
    // Se realiza el cambio solicitado
    return ereg_replace("Línea",
        "<font color='red'>Línea</font>",
        $cadena);
}

define('NOMBRE_FICHERO', 'ejemplo.html');
$fichero = fopen(NOMBRE_FICHERO, 'r') or
    die('Error de apertura');
$num_linea = 0;
$captura = true;
// Se captura la salida estándar y se almacena
    en un búffer
ob_start("tratar");
```

Lee 10 líneas de un fichero, las almacena en un buffer y antes de enviarlas a la salida estándar las cambia a color rojo.

```
while (!feof($fichero)) {
    // Se leen las líneas del fichero y se elabora
    la respuesta
    $num_linea++;
    $linea = fgetss($fichero, 4096, '<body>
        <hr>');
    // Se escribe sobre el búffer
    echo "Línea $num_linea: $linea<br />\n";
    // Se comprueba si se ha llegado a la décima
    línea
    if ($num_linea >= 10 and $captura) {
        // Se ha llegado a la décima fila y se
        termina la captura del búffer.
        $captura = false;
        ob_end_flush();
    }
}
fclose($fichero);
// Se finaliza la captura si el fichero leído no
    tenía diez líneas
if ($captura)
    ob_end_flush();
?>
```

Directivas

- **allow_url_open**: indica si fopen puede trabajar con urls.
- **default_socket_timeout**: Tiempo máximo permitido para realizar operaciones sobre ficheros a los que se accede mediante sockets.
- **auto_detect_line_endings**: Permite a PHP examinar los datos leídos de fgets() y file() para saber si están usando convenciones sobre sistemas win, unix, mac. Baja el rendimiento, solo usarla para sistemas mac.
- **output_buffering**: on / off. Se activa o no la técnica buffering para todos los ficheros o no.

Gestión de Directorios

- Cambio de dir: `boolean chdir(string ruta)`
 - Sobre Windows podemos utilizar `\` o `/`.
 - Sobre Unix solo `/`.
- Creación:
 - `boolean mkdir(ruta, permisos);`
- Borrado:
 - `boolean rmdir(ruta);`

Recorrer un directorio

```
$manejador = opendir('');  
echo "Elementos del directorio actual: ";  
while ($elemento = readdir($manejador))  
    echo "$elemento" . "<br />";  
  
closedir($manejador);
```