Unidad 3

Ficheros y directorios

Contacto: consultas@elearning-total.com

Web: www.elearning-total.com

FUNCIONES PARA EL MANEJO DE ARCHIVOS

Abrir Archivos

Para abrir archivos, se utiliza la función **fopen**, cuya sintaxis es la siguiente:

fopen (nombre_archivo, modo);

En donde, nombre_archivo es el nombre del archivo que se quiere abrir o crear y el modo indica de qué forma se procederá a la apertura del archivo. Los distintos modos se comentan en los ejemplos siguientes:

fopen (archivo.txt, 'a')

Abre el archivo en modo de agregar información. Los datos que se ingresen se agregarán al final del mismo, sin eliminar el contenido que tuviera. En este modo, si el archivo no existe, lo creará en ese momento.

fopen (archivo.txt, 'a+')

Abre el archivo en modo de agregar información y además leerlo. Los datos que se ingresen se agregarán al final del mismo, sin eliminar el contenido que tuviera. En este modo, si el archivo no existe, lo creará en ese momento.

fopen (archivo.txt, 'r')

Abre el archivo en modo lectura solamente. El archivo debe existir.

fopen (archivo.txt, 'r+')

Abre el archivo en modo lectura y escritura. La información que se agregue, será insertada al principio del archivo.

fopen (archivo.txt, 'w')

Abre el archivo en modo escritura solamente. Si el archivo no existe, lo crea, y si existe con algún contenido, elimina toda su información, dejándolo en blanco.

fopen (archivo.txt, 'w+')

Abre el archivo en modo escritura y lectura. Si el archivo no existe, lo crea, y si existe con algún contenido, elimina toda su información, dejándolo en blanco.

Cerrar Archivos

Luego de abrir un archivo y realizar las operaciones necesarias en él, se debe cerrarlo. Para cerrar un archivo se utiliza la función **fclose()** que recibe como parámetro la variable del archivo que se está utilizando.

Ejemplo:

```
$f = fopen("archi.txt", ‘r’);
fclose($f);
```

Escritura en un Archivo

Para guardar información en un archivo de texto se utilizan las funciones **fputs()** y **fwrite(),** que se detallan a continuación:

fputs()

Permite escribir en un archivo. Recibe tres parámetros, de los cuales los dos primeros son obligatorios y el tercero es opcional. La sintaxis es la siguiente:

fputs (variable_fichero, texto, largo)

El primer parámetro es el puntero al archivo, es decir la variable de trabajo. El segundo parámetro es el texto que se desea escribir. El tercer parámetro es el largo de la cadena, si no se coloca, se grabará la cadena entera. Ejemplo: El siguiente ejemplo muestra un pequeño programa que graba una cadena de texto en un archivo llamado Texto1.txt

```
<html>
<head>
<title> ejemplo fputs </title>
</head>
<body>
<?php
$texto="la materia se transforma consumiendo o liberando energía.";
$f = fopen("texto1.txt",”w”);
fputs($f, $texto);
echo "texto almacenado correctamente"; fclose($f);
?>
</body>
</html>
```

Al mismo resultado se hubiera llegado utilizando la función fwrite(), es decir, si en el ejemplo anterior se reemplaza fputs() por fwrite(), el resultado del programa no se alteraría.

Verificar una vez ejecutado el archivo desde el localhost si en la carpeta donde se encuentra el archivo ejemplo01-fputs.php se creó el archivo texto1.txt con el texto:

"la materia se transforma consumiendo o liberando energía."

Lectura de un Archivo

Para poder ver el contenido de un archivo de texto se pueden utilizar varias funciones:

fpassthru()

Permite ver el contenido completo de un archivo. Tiene como parámetro la variable archivo utilizada para abrirlo.

Ejemplo: El siguiente ejemplo lee el archivo de texto grabado en el ejemplo 1.

```
<html>
<head>
<title> ejemplo fpassthu </title>
</head>
<body>
<?php
$nombre="texto1.txt";
$f = fopen($nombre,"r") or die("error al abrir el archivo: $nombre"); echo "texto:" . "<br/>";
fpassthru($f);
echo "<br/><br/>";
echo "el texto se ha leído correctamente";
fclose($f);
?>
</body>
</html>
```

En primera instancia, se abre el archivo en modo lectura (r) y se utiliza la instrucción die() para poder imprimir un mensaje de error cuando el fichero a leer no exista. Con la instrucción fpassthru() se lee y se muestra el archivo.

fread()

Permite leer parte de un archivo abierto. Esta función tiene dos parámetros y su sintaxis es la siguiente:

fread(variable_fichero, cantidad);

variable_fichero: es el puntero al archivo abierto de trabajo. cantidad: es la cantidad de caracteres que se pretende leer.

Ejemplo:

En el próximo ejemplo se leen los primeros 24 caracteres del texto grabado en el ejemplo 1.

```
<html>
<head>
<title> ejemplo fread </title>
</head>
<body>
<?php
$nombre="texto1.txt";
$f = fopen($nombre,”r”) or die("error al abrir el archivo: $nombre");
echo "<h3>";
echo "texto leido:" . "<br/>"; echo fread($f,24);
echo "<br/><br/>";
echo "se han leído los 24 primeros caracteres"; fclose($f);
echo "</h3>";
?>
</body>
</html>
```

fgetc()

Permite leer un archivo de texto carácter por carácter, también se puede recorrer el archivo parcial o totalmente.

Ejemplo:

En este ejemplo se lee el archivo completo utilizando la función fgetc(), mediante un ciclo leyendo carácter por carácter.

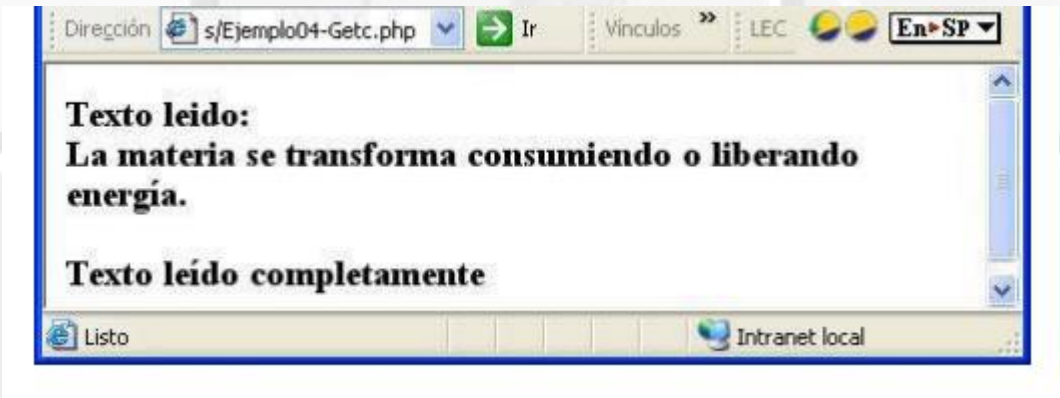
```
<html>
<head>
<title> ejemplo04-getc.php </title>
</head>
<body>
<?php
$nombre="texto1.txt";
$f = fopen($nombre,”r”) or die("error al abrir el archivo: $nombre"); echo "<h3>";
echo "texto leido:" . "<br/>"; while (!feof($f))
{
echo fgetc($f);
}
echo "<br/><br/>";
echo "texto leído completamente";
fclose($f); echo "</h3>";
?>
</body>
</html>
```

Para el ciclo se utilizó la estructura while(), que tiene como condición a la función feof(). Esta función significa fin de fichero (end of file), y se encarga de indicar si se llegó al final del archivo o no. En el ejemplo, la función está negada con el símbolo de admiración delante, por lo tanto, la línea se lee de la siguiente manera:

Mientras no sea el fin del fichero, procese.

Lo que procesará será la función fgetc(), que leerá caracteres hasta que se termine en archivo, saliendo del ciclo while().

El resultado de este ejemplo es el siguiente:



fgets()

Permite leer una cadena de texto de un archivo, tiene dos parámetros y su sintaxis es la siguiente:

fgets (variableFichero, longitud);

variable_Fichero: es el puntero al archivo abierto de trabajo. longitud: Es opcional, y es la cantidad de caracteres – 1, que se pretende leer. Es decir que si se pretende leer 24 caracteres por ejemplo, la longitud debería ser de 25.

Ejemplo:

```
<html>
<head>
<title> ejemplo05-fgets.php </title>
</head>
<body>
<?php
```

```
$nombre="texto1.txt";
$f = fopen($nombre,”r”) or
die("error al abrir el archivo: $nombre"); echo "<h3>";
echo "texto leido:" . "<br/>"; echo fgets($f,25);
fclose($f); echo "</h3>";
?>
</body>
</html>
```

file()

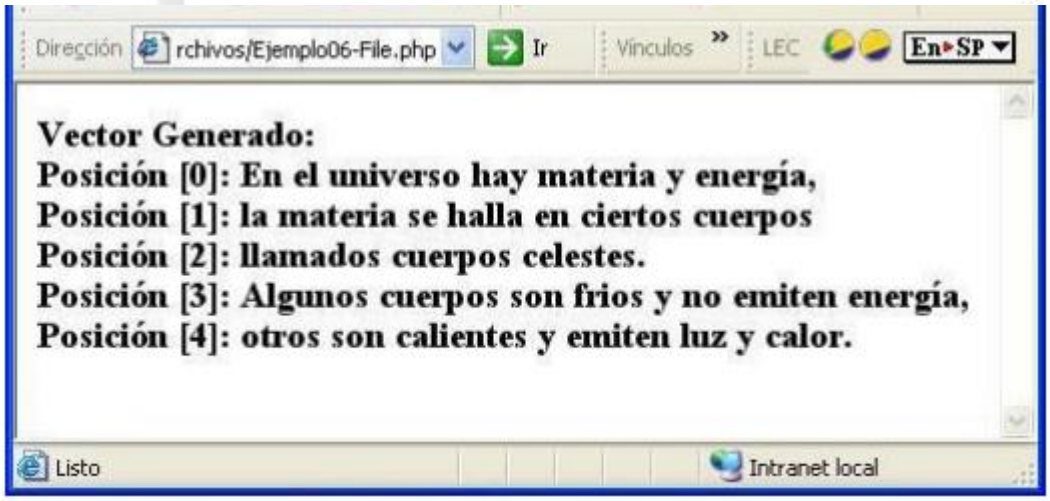
Esta función permite almacenar un texto en un vector. El almacenamiento se producirá línea por línea, es decir que cada elemento del vector contendrá una línea de texto.

Ejemplo:

Guardar en un archivo, un determinado texto y posteriormente almacenarlo línea por línea en un vector.

```
<html>
<head>
<title> ejemplo06-file.php </title>
</head>
<body>
<?php
// almacenamiento del texto
$texto="en el universo hay materia y energía, la materia se halla en ciertos cuerpos llamados
cuerpos celestes. Algunos cuerpos son frios y no emiten energía, otros son calientes y emiten
luz y calor.";
$f = fopen("texto2.txt",”w”);
fwrite($f, $texto); fclose($f);
// generación del vector
$nombre="texto2.txt";
$f = fopen($nombre,”r”) or die("error al abrir el archivo: $nombre"); fclose($f);
$v = file($nombre);
$cantidad = count($v); echo "<h3>";
echo "vector generado:" . "<br/>";
for ($i=0; $i<$cantidad; $i++)
{
echo "posición [$i]: $v[$i]" . "<br/>";
}
echo "</h3>";
```

```
?>
</body>
</html>
```



Recorrido de un Archivo

Las siguientes funciones muestran como se posiciona en el archivo en un lugar determinado, leer una porción de texto y tomar la posición de donde se encuentra el puntero en el archivo.

rewind()

La función rewind() mueve el puntero al inicio del archivo. Recibe como parámetro a la variable fichero que se abrió previamente.

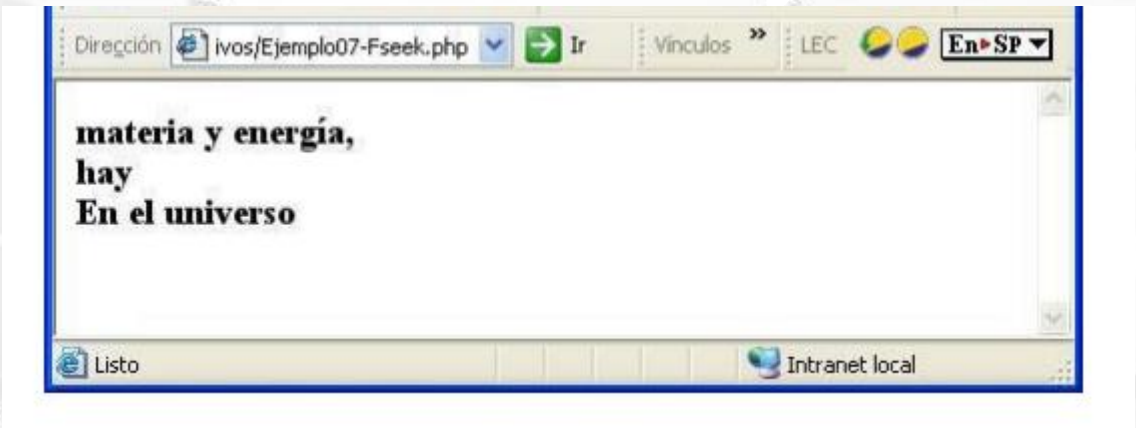
fseek()

La función fseek() permite situar el puntero en un lugar específico. Tiene dos parámetros, el primero es la variable fichero, y el segundo es el número de carácter donde se quiere posicionar. (Tener presente de que el primer carácter es el número 0).

Ejemplo:

El siguiente ejemplo, se posiciona en el carácter número 19 y lee los 18 caracteres siguientes del Texto2.txt, luego en el próximo renglón, se posiciona en el carácter número 15 y lee los siguientes 4 caracteres y por último, se posiciona al principio con la función rewind() y se leen los primeros 14 caracteres.

```
<html>
<head>
<title> ejemplo fseek </title>
</head>
<body>
<?php
$nombre="texto2.txt";
$f = fopen($nombre,”r”) or die("error al abrir el archivo: $nombre"); echo "<h3>";
fseek($f, 19);
echo fread($f, 18); echo "<br/>"; fseek($f, 15);
echo fread($f, 4); echo "<br/>"; rewind($f);
echo fread($f, 14); fclose($f);
echo "<h3>";
?>
</body>
</html>
```



ftell()

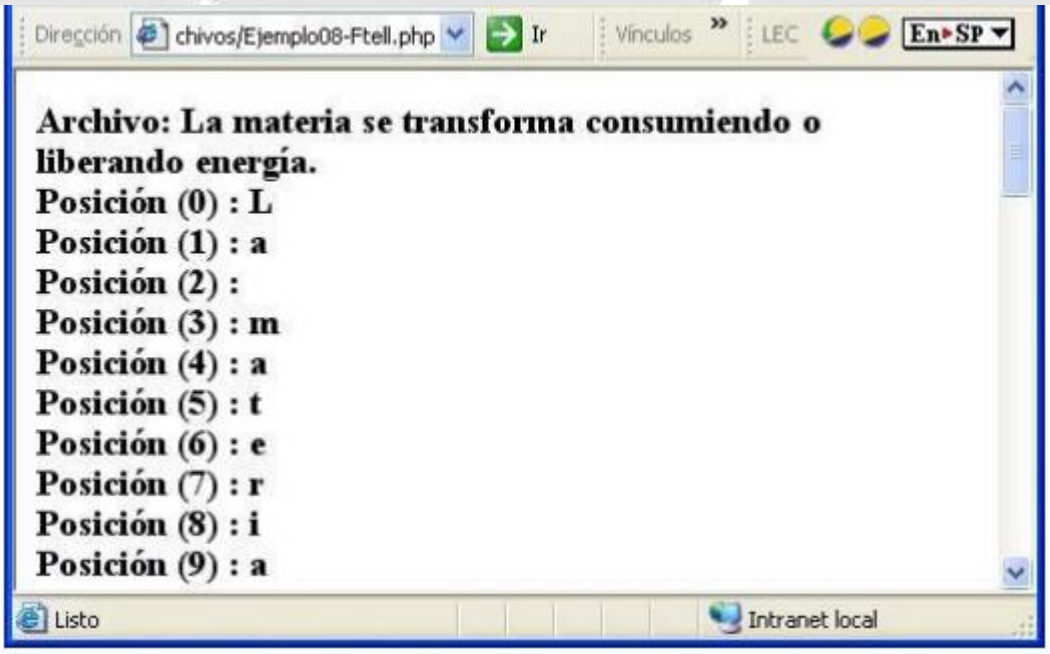
Permite obtener la posición de donde está el puntero en el archivo. Recibe como parámetro la variable fichero y devuelve el valor numérico de la posición en que está el puntero.

Ejemplo:

Leer un archivo y mostrar las posiciones de cada uno de los caracteres que lo constituyen.

```
<html>
<head>
<title> ejemplo08-ftell.php </title>
</head>
```

```
<body>
<?php
$nombre="texto1.txt";
$f = fopen($nombre,”r”) or die("error al abrir el archivo: $nombre");
echo "<h3>";
echo "archivo: "; echo fgets($f); echo "<br/>"; rewind($f); while (!feof($f))
{
echo " posición (" . ftell($f) . ") : "; echo fgetc($f);
echo "<br>";
}
fclose($f);
echo "<h3>";
?>
</body>
</html>
```



Funciones Útiles para el manejo de Archivos

Se describen a continuación funciones útiles para el manejo de archivos, entre ellas se verá como copiar, renombrar y obtener información de los archivos almacenados.

copy()

Esta función permite realizar una copia de un archivo determinado. Necesita dos parámetros, que son: el nombre del archivo origen y el nombre del archivo destino. Si la copia se realiza con éxito devuelve verdadero, de lo contrario devuelve falso.

Ejemplo:

Realizar una copia del archivo Texto1.txt en el archivo Texto1.bak

```
<html>
<head>
<title> ejemplo copy </title>
</head>
<body>
<?php
$origen="texto1.txt";
$f = fopen($origen,”r”) or die("error al abrir el archivo: $origen"); echo "<h3>";
echo "archivo: "; echo fgets($f); echo "<br/>"; fclose($f);
$destino="texto1.bak";
if (copy($origen, $destino))
{
echo "la copia se realizó con éxito";
}
else
{
echo "no se pudo realizar la copia";
}
echo "</h3>";
?>
</body>
</html>
```

file_exists()

La función file_exists() permite determinar la existencia o no del archivo pasado como parámetro. Devuelve verdadero en caso de existir el archivo, en caso contrario devuelve falso. Generalmente se utiliza esta función cuando se pretende borrar el archivo, renombrarlo o simplemente abrirlo.

Ejemplo:

Verificar si el archivo Texto1.txt existe o no.

```
<html>
<head>
<title> ejemplo file_exists </title>
</head>
<body>
<?php
$nombre="texto1.txt";
if (file_exists($nombre))
{
echo "el archivo $nombre existe";
}
else
{
echo "el archivo $nombre no existe";
}
?>
</body>
</html>
```

rename()

Permite cambiar el nombre a un archivo. Acepta dos parámetros, el primero es el nombre actual del archivo y el segundo es el nombre nuevo que se le quiere dar.

Ejemplo:

Cambiar el nombre del archivo Texto1.bak por Archi1.txt.

```
<html>
<head>
<title> ejemplo rename </title>
</head>
```

```
<body>
<?php
$nombreactual="texto1.bak";
$nombrenuevo ="archi1.txt"; if (file_exists($nombreactual))
{
rename($nombreactual, $nombrenuevo);
echo "el nombre se cambió correctamente";
}
else
{
echo "no se encontro el archivo $nombreactual";
}
?>
</body>
</html>
```

unlink()

Esta función se utiliza para borrar un archivo. Recibe como parámetro el nombre del archivo a borrar.

Ejemplo 12:

Borrar el archivo Archi1.txt.

```
<html>
<head>
<title> ejemplo12-unlink.php </title>
</head>
<body>
<?php
$nombre="archi1.txt"; if (file_exists($nombre))
{
unlink($nombre);
echo "el archivo $nombre se borró correctamente";
}
else
{
echo "no se encontro el archivo $nombre";
}
?>
</body>
```

```
</html>
```

fileatime() y filemtime()

La función fileatime() devuelve la última fecha de acceso a un archivo determinado y la función filemtime(), devuelve la última fecha de modificación de un archivo. Ambas reciben como parámetro el nombre del archivo.

Ejemplo:

Mostrar las últimas fechas de acceso y de modificación del archivo Texto2.txt.

```
<html>
<head>
<title> ejemplo filetime </title>
</head>
<body>
<?php
$nombre="texto2.txt"; echo "<h3>";
if (file_exists($nombre))
{
echo "archivo: $nombre" ; echo "<br/><br/>";
echo "última fecha de acceso: ";
echo date("d/m/y",fileatime($nombre)); echo "<br/><br/>";
echo "última fecha de modificación: ";
echo date("d/m/y",filemtime($nombre));
}
else
{
echo "no se encontro el archivo $nombre";
}
echo "</h3>";
?>
</body>
</html>
```

En este ejemplo también se utilizó la función date(), que como ya hemos visto, tiene por finalidad darle un formato a la fecha mostrada. Tiene dos parámetros, el primero es el formato y el segundo es un número entero, que en este caso es el valor devuelto por las funciones fileatime() y filemtime().

filesize()

Esta función devuelve el tamaño del archivo que se le pasa como parámetro. El valor devuelto es en cantidad de caracteres.

Ejemplo:

Determinar la cantidad de caracteres que contiene el archivo llamado Texto1.txt.

```
<html>
<head>
<title> ejemplo filesize </title>
</head>
<body>
<?php
$nombre="texto1.txt"; echo "<h3>";
if (file_exists($nombre))
{
echo "tamaño de $nombre: "; echo filesize($nombre);
}
else
{
echo "no se encontro el archivo $nombre";
}
echo "</h3>";
?>
</body>
</html>
```

Move_uploaded_file()

bool move_uploaded_file (string $filename , string $destination)

La función move_uploaded_file espera recibir dos parámetros:

filename

El nombre de archivo del archivo subido.

destination

El destino del archivo movido.

Y devuelve true si la subida del archivo se hizo correctamente o false en caso de que haya ocurrido algún error.

En la carpeta subearchivo tenemos un ejemplo de uso de esta función.

Quiero destacar dos cosas del script formulario_archivo.php.

En primer lugar tenemos un input type=”file”, para que este dato pase correctamente al php definido en el action de nuestra etiqueta <form> tenemos que indicar en dicha etiqueta un enctype diferente: **enctype="multipart/form-data"** .

Hasta ahora no indicábamos nada en el enctype porque automáticamente asumía el enctype por defecto que es application/x-www-form-urlencoded y que con los input type=”text” que veníamos usando nos era suficiente.

Y en segundo lugar este input type=”file” genera una variable $_FILES que es la que vamos a tener que tomar en el archivo subearchivo.php

Vemos el código de nuestro script formulario_archivo.php:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Ejemplo</title>
</head>
<body>
<form action="subearchivo.php" method="post" enctype="multipart/form-data">
 <b>Campo de tipo texto:</b>
 <br>
 <input type="text" name="cadenatexto" size="20" maxlength="100">
 <input type="hidden" name="MAX_FILE_SIZE" value="100000">
 <br>
 <br>
 <b>Enviar un nuevo archivo: </b>
 <br>
 <input name="userfile" type="file">
```

```
 <br>
 <input type="submit" value="Enviar">
</form>
</body>
</html>
```

Esta variable $_FILES es un array compuesto de 5 subíndices:

name: es el nombre del archivo a subir
type: el tipo
size: el tamaño
tmp_name: el nombre del archive temporal que se usara en la subida
error: indica un codigo en caso de error

Y esos son los datos que usamos en el srcript subearchivo.php para subir el archivo al servidor. En este caso también estamos controlando que tenga una extensión específica y un tamaño menor a 100000 bytes.

Esto es arbitrario si quieren aceptar la subida de cualquier tipo de archivo eliminan el if con la condición y listo.

Si quieren ver el contenido de la variable $_FILES pueden descomentar del código el foreach que está comentado.

```
<?php
// foreach ($_FILES['userfile'] as $indice=>$valor){
// echo 'indice '.$indice.'valor '.$valor;
// }
//tomo el valor de un elemento de tipo texto del formulario
$cadenatexto = $_POST["cadenatexto"];
echo "Escribi&oacute; en el campo de texto: " . $cadenatexto . "<br><br>";
//datos del arhivo
$nombre_archivo = $_FILES['userfile']['name'];
$tipo_archivo = $_FILES['userfile']['type'];
$tamano_archivo = $_FILES['userfile']['size'];
//compruebo si las características del archivo son las que deseo
if (($tipo_archivo != 'image/jpeg' && $tipo_archivo != 'image/gif' && $tipo_archivo
!='image/png') && ($tamano_archivo < 100000)){
 echo "La extensi&oacute;n o el tamaño de los archivos no es correcta.
<br><br><table><tr><td><li>Se permiten archivos .gif o .jpg</li><li>se permiten archivos de
100 Kb m&aacute;ximo.</li></td></tr></table>";
```

```
}else{
 if (move_uploaded_file($_FILES['userfile']['tmp_name'], $nombre_archivo)){
 echo "El archivo ha sido cargado correctamente.";
 }else{
 echo "Ocurri&oacute; alg&uacute;n error al subir el fichero. No pudo guardarse.";
 }
}
?>
```