Capitulo 6: Manejo de Archivos.

1. Apertura de un archivo.

La función utilizada para abrir un archivo en PHP es fopen, la sintaxis.

fp_handler=fopen("path","modo");

Parh es la ruta completa del archivo a abrir, si el path comienza con "http://" se realiza una conexión a la URL indicada y se abre la página como si fuera un archivo (con las limitaciones lógicas, por ejemplo no es posible escribir).

Los modos en los que se puede abrir un archivo son:

r	Sólo lectura
r+	Lectura y escritura
w	Sólo escritura, si no existe el archivo lo crea, si existe lo trunca
w+	Lectura y escritura, si existe lo trunca, si no existe lo crea
a	Modo append sólo escritura si no existe lo crea
a+	Modo append lectura y escritura si no existe lo crea

La función devuelve un file_handler que luego debe ser usado en todas las funciones de tipo fnombre_funcion, como por ejemplo fgets, fputs, fclose, fread, fwrite, etc.

2. Lectura desde un archivo.

Las funciones que pueden usarse para leer un archivo son:

string=**fgets**(file_handler, longitud)

Lee una línea de texto hasta el fin de línea o bien hasta que se cumpla la longitud indicada, devuelve el resultado en la variable pasada. El archivo debe estar abierto con fopen.

var=**fread**(file_handler, cantidad)

Lee la cantidad de bytes indicados ignorando saltos de línea y deja el resultado en la variable var.

Ejemplo

$buffer=**fread**($fp, 1024); //Lee 1Kb desde el archivo cuyo handler es $fp

string=**fgetss**(file_handler, longitud)

Idéntica a fgets con la diferencia de que los tags html son eliminados del archivo a medida que se lee el mismo. Opcionalmente puede pasarse una lista de tags que no deben ser eliminados.

Ejemplo:

$string=fgetss($fp,999999,"<b> <i> <table> <tr> <td>");

Lee una línea (de cualquier longitud) eliminando los tags html excepto los indicados como segundo parámetro. Los tags que cierran los tags especificados en la lista de tags permitidos tampoco son eliminados.

3. Escritura a un archivo

fwrite(file_handler, variable, longitud);

Escribe la variable al archivo indicado por file_handler. Si esta indicado el parámetro "longitud" (que es opcional) se escribirán tantos bytes como la longitud indicada por dicho parámetro o como la longitud de la variable, en aquellos casos en que el parámetro longitud es mayor que la longitud de la variable.

La función devuelve la cantidad de bytes escritos en el archivo.

Ejemplo:

```
$q = fwrite($fp,$buffer,999999);
```

fputs es idéntico a fwrite y funciona de la misma manera. (es un alias).

4. Cierre de archivos

fclose(file_handler)

Cierra un archivo abierto con fopen.

5. Fin de archivo

boolean = feof(file_handler);

Devuelve verdadero si no quedan más bytes para leer en el archivo o si se produce algún tipo de error al leer.

Ejemplo:

```
$fp=fopen("/usr/luis/archivo.txt","r");
while(!feof($fp)) {
  $s=fgets($fp,999999);
  print("$s");
}
```

6. Manejo de archivos.

PHP provee funciones para copiar, renombrar, mover y borrar archivos y directorios, las funciones son:

Función	Descripción
rename(path_origen, path_destino);	Renombra un archivo. Ejemplo: $newname="/usr/eduardo/file.txt"; Rename("/usr/eduardo/archivo.txt","$newname");
unlink(path_a_borrar);	Elimina un archivo.
rmdir(directorio_a_borrar);	Elimina un directorio (debe estar vacío)
mkdir(path_a_crear);	Crea un directorio Nuevo.
copy(path_origen, path_destino);	Copia un archivo o varios.

7. Otras funciones útiles.

Función	Descripción
fseek(file_handler,posicion,desde)	Posiciona el puntero de lectura/escritura de un archivo en el offset indicado. El parámetro "desde" es opcional y puede tomar uno de los siguientes valores: SEEK_SET (el offset es absoluto) SEEK_CUR (el offset es relativo a la posición actual) SEEK_END (el offset es desde el final del archivo) El default es SEEK_SET
ftruncate(file_handler, longitud)	Trunca el archivo indicado a la longitud en bytes especificada.
array=file(path)	Lee un archivo de texto y devuelve un vector donde cada elemento del vector es una línea del archivo.
file_exists(path)	Devuelve true/false según el path indicado exista o no.
filemtime(path)	Devuelve la fecha de última modificación de un archivo en formato Unix. (ver manejo de fechas)
filesize(path)	Devuelve el tamaño de un archivo.
filetype(path)	Devuelve el tipo de un archivo.
flock(file_handler,modo)	Lockea un archivo (independientemente del file-system), el modo puede ser: 1: Lock en modo lectura (compartido) 2: Lock en modo escritura (exclusivo) 3: Release del lock adquirido. Al hacer un fclose del archivo se liberan automáticamente los locks adquiridos sobre el mismo. Si flock no puede obtener el lock espera hasta que el lock este disponible, si se quiere que flock no bloquee el script sumar 4 al modo (modos: 5,6,7) y consultar por el valor devuelto por la función: true si el lock fue adquirido o false si no fue adquirido. Usando esta función pueden implementarse mecanismos de sincronización entre procesos.
fpassthru(file_handler)	Escribe al standard_output el contenido del archivo.
readfile(path)	Lee todo el archivo y lo escribe al standard output.
is_dir(path)	True/false según el path sea un directorio
is_file(path)	True/false según el path sea un archivo
tempnam(path)	Dado el nombre de un directorio devuelve un nombre de archivo que no existe en el directorio dado (útil para crear archivos temporarios)

Manejo de directorios.

Las siguientes funciones de PHP están orientadas a recorrer directorios y obtener los archivos que se encuentran dentro de ellos. Las funciones son independientes del file-system con lo cual funcionan correctamente tanto en UNIX como en Windows.

directory_handle = opendir(path);

Abre el directorio pasado como parámetro y devuelve un handler al directorio, previamente puede usarse la función is_dir(path) para verificar si el path es un directorio en caso de que esta validación sea necesaria.

string = readdir(directory_handler)

Lee la próxima entrada en el directorio abierto con opendir, usualmente las dos primeras entradas de un directorio con "." y ".." por lo que el código que procesa los archivos del directorio debe tener esto en cuenta para no procesar dichas entradas en caso de que tal cosa no sea deseable.

closedir(directory_handler)

Cierra un handler abierto con opendir.

rewinddir(directory_handler)

Rebobina el puntero interno de un directorio para que apunte nuevamente al comienzo del mismo.