 **Funciones de Archivos**

***© Julio de 2001, Salvador Pozo Coronado,***[***salvador@conclase.net***](mailto:salvador@conclase.net)

[**1 Generalidades**](http://c.conclase.net/ficheros/?cap=001#inicio)

[**2 Funciones C para ficheros**](http://c.conclase.net/ficheros/?cap=002#inicio)

* [***Funciones y tipos***](http://c.conclase.net/ficheros/?cap=002#FIC_FUNCC)
* [***Funciones C para acceso aleatorio***](http://c.conclase.net/ficheros/?cap=002#FIC_FUNCC2)
* [***Clases para manejar ficheros***](http://c.conclase.net/ficheros/?cap=002b#FIC_FUNCCPP)

[***Métodos para acceso aleatorio en C++***](http://c.conclase.net/ficheros/?cap=002b#FIC_FUNCCPP2)

[**3 Archivos secuenciales**](http://c.conclase.net/ficheros/?cap=003#inicio)

[**4 Archivos de acceso aleatorio**](http://c.conclase.net/ficheros/?cap=004#inicio)

[**5 Ordenar archivos secuenciales**](http://c.conclase.net/ficheros/?cap=005#inicio)

[**6 Ordenar archivos de acceso aleatorio**](http://c.conclase.net/ficheros/?cap=006#inicio)

[**7 Ficheros de índices**](http://c.conclase.net/ficheros/?cap=007#inicio)

[**8 Ficheros indicados no ordenados**](http://c.conclase.net/ficheros/?cap=008#inicio)

[**9 Ficheros indicados Árboles-B**](http://c.conclase.net/ficheros/?cap=009#inicio)

**Función [fopen](http://c.conclase.net/librerias/?ansifun=fopen):**

Sintaxis:

FILE \*fopen(char \*nombre, char \*modo);

Esta función sirve para abrir y crear ficheros en disco. El valor de retorno es un puntero a una estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE). Los parámetros de entrada son:

1. nombre: una cadena que contiene un nombre de fichero válido, esto depende del sistema operativo que estemos usando. El nombre puede incluir el camino completo.
2. modo: especifica en tipo de fichero que se abrirá o se creará y el tipo de datos que puede contener, de texto o binarios:
   * r: sólo lectura. El fichero debe existir.
   * w: se abre para escritura, se crea un fichero nuevo o se sobreescribe si ya existe.
   * a: añadir, se abre para escritura, el cursor se situa al final del fichero. Si el fichero no existe, se crea.
   * r+: lectura y escritura. El fichero debe existir.
   * w+: lectura y escritura, se crea un fichero nuevo o se sobreescribe si ya existe.
   * a+: añadir, lectura y escritura, el cursor se situa al final del fichero. Si el fichero no existe, se crea.
   * t: tipo texto, si no se especifica "t" ni "b", se asume por defecto que es "t"
   * b: tipo binario.

**Función [fclose](http://c.conclase.net/librerias/?ansifun=fclose):**

Sintaxis:

int fclose(FILE \*fichero);

Es importante cerrar los ficheros abiertos antes de abandonar la aplicación. Esta función sirve para eso. Cerrar un fichero almacena los datos que aún están en el buffer de memoria, y actualiza algunos datos de la cabecera del fichero que mantiene el sistema operativo. Además permite que otros programas puedan abrir el fichero para su uso. Muy a menudo, los ficheros no pueden ser compartidos por varios programas.

Un valor de retorno cero indica que el fichero ha sido correctamente cerrado, si ha habido algún error, el valor de retorno es la constante EOF. El parámetro es un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero que queremos cerrar.

**Función [fgetc](http://c.conclase.net/librerias/?ansifun=fgetc):**

Sintaxis:

int fgetc(FILE \*fichero);

Esta función lee un carácter desde un fichero.

El valor de retorno es el carácter leído como un **unsigned char** convertido a **int**. Si no hay ningún carácter disponible, el valor de retorno es EOF. El parámetro es un puntero a una estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero del que se hará la lectura.

**Función [fputc](http://c.conclase.net/librerias/?ansifun=fputc):**

Sintaxis:

int fputc(int caracter, FILE \*fichero);

Esta función escribe un carácter a un fichero.

El valor de retorno es el carácter escrito, si la operación fue completada con éxito, en caso contrario será EOF. Los parámetros de entrada son el carácter a escribir, convertido a **int** y un puntero a una estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero en el que se hará la escritura.

**Función [feof](http://c.conclase.net/librerias/?ansifun=feof):**

Sintaxis:

int feof(FILE \*fichero);

Esta función sirve para comprobar si se ha alcanzado el final del fichero. Muy frecuentemente deberemos trabajar con todos los valores almacenados en un archivo de forma secuencial, la forma que suelen tener los bucles para leer todos los datos de un archivo es permanecer leyendo mientras no se detecte el fin de fichero. Esta función suele usarse como prueba para verificar si se ha alcanzado o no ese punto.

El valor de retorno es distinto de cero sólo si no se ha alcanzado el fin de fichero. El parámetro es un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero que queremos verificar.

### Función [fgets](http://c.conclase.net/librerias/?ansifun=fgets):

Sintaxis:

char \*fgets(char \*cadena, int n, FILE \*fichero);

Esta función está diseñada para leer cadenas de caracteres. Leerá hasta n-1 caracteres o hasta que lea un retorno de línea. En este último caso, el carácter de retorno de línea también es leído.

El parámetro *n* nos permite limitar la lectura para evitar derbordar el espacio disponible en la *cadena*.

El valor de retorno es un puntero a la cadena leída, si se leyó con éxito, y es [NULL](http://c.conclase.net/librerias/?ansimac=tNULL) si se detecta el final del fichero o si hay un error. Los parámetros son: la cadena a leer, el número de caracteres máximo a leer y un puntero a una estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero del que se leerá.

### Función [fputs](http://c.conclase.net/librerias/?ansifun=fputs):

Sintaxis:

int fputs(const char \*cadena, FILE \*stream);

La función [fputs](http://c.conclase.net/librerias/?ansifun=fputs) escribe una cadena en un fichero. No se añade el carácter de retorno de línea ni el carácter nulo final.

El valor de retorno es un número no negativo o EOF en caso de error. Los parámetros de entrada son la cadena a escribir y un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero donde se realizará la escritura.

### Función [fread](http://c.conclase.net/librerias/?ansifun=fread):

Sintaxis:

size\_t fread(void \*puntero, size\_t tamaño, size\_t nregistros, FILE \*fichero);

Esta función está pensada para trabajar con registros de longitud constante. Es capaz de leer desde un fichero uno o varios registros de la misma longitud y a partir de una dirección de memoria determinada. El usuario es responsable de asegurarse de que hay espacio suficiente para contener la información leída.

El valor de retorno es el **número de registros leídos**, no el número de bytes. Los parámetros son: un puntero a la zona de memoria donde se almacenarán los datos leídos, el tamaño de cada registro, el número de registros a leer y un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero del que se hará la lectura.

### Función [fwrite](http://c.conclase.net/librerias/?ansifun=fwrite):

Sintaxis:

size\_t fwrite(void \*puntero, size\_t tamaño, size\_t nregistros, FILE \*fichero);

Esta función también está pensada para trabajar con registros de longitud constante y forma pareja con fread. Es capaz de escribir hacia un fichero uno o varios registros de la misma longitud almacenados a partir de una dirección de memoria determinada.

El valor de retorno es el **número de registros escritos**, no el número de bytes. Los parámetros son: un puntero a la zona de memoria donde se almacenarán los datos leídos, el tamaño de cada registro, el número de registros a leer y un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero del que se hará la lectura.

Ejemplo:

// copia.c: Copia de ficheros

// Uso: copia <fichero\_origen> <fichero\_destino>

**#include** <stdio.h>

**int** *main*(**int** argc, **char** \*\*argv) {

[FILE](http://c.conclase.net/librerias/?ansitip=FILE) \*fe, \*fs;

**unsigned** **char** buffer[2048]; // Buffer de 2 Kbytes

**int** bytesLeidos;

**if**(argc != 3) {

[printf](http://c.conclase.net/librerias/?ansifun=printf)("Usar: copia <fichero\_origen> <fichero\_destino>\n");

**return** 1;

}

// Abrir el fichero de entrada en lectura y binario

fe = [fopen](http://c.conclase.net/librerias/?ansifun=fopen)(argv[1], "rb");

**if**(!fe) {

[printf](http://c.conclase.net/librerias/?ansifun=printf)("El fichero %s no existe o no puede ser abierto.\n", argv[1]);

**return** 1;

}

// Crear o sobreescribir el fichero de salida en binario

fs = [fopen](http://c.conclase.net/librerias/?ansifun=fopen)(argv[2], "wb");

**if**(!fs) {

[printf](http://c.conclase.net/librerias/?ansifun=printf)("El fichero %s no puede ser creado.\n", argv[2]);

[fclose](http://c.conclase.net/librerias/?ansifun=fclose)(fe);

**return** 1;

}

// Bucle de copia:

**while**((bytesLeidos = [fread](http://c.conclase.net/librerias/?ansifun=fread)(buffer, 1, 2048, fe)))

[fwrite](http://c.conclase.net/librerias/?ansifun=fwrite)(buffer, 1, bytesLeidos, fs);

// Cerrar ficheros:

[fclose](http://c.conclase.net/librerias/?ansifun=fclose)(fe);

[fclose](http://c.conclase.net/librerias/?ansifun=fclose)(fs);

**return** 0;

}

### Función [fprintf](http://c.conclase.net/librerias/?ansifun=fprintf):

Sintaxis:

int fprintf(FILE \*fichero, const char \*formato, ...);

La función [fprintf](http://c.conclase.net/librerias/?ansifun=fprintf) funciona igual que [printf](http://c.conclase.net/librerias/?ansifun=printf) en cuanto a parámetros, pero la salida se dirige a un fichero en lugar de a la pantalla.

### Función [fscanf](http://c.conclase.net/librerias/?ansifun=fscanf):

Sintaxis:

int fscanf(FILE \*fichero, const char \*formato, ...);

La función [fscanf](http://c.conclase.net/librerias/?ansifun=fscanf) funciona igual que [scanf](http://c.conclase.net/librerias/?ansifun=scanf) en cuanto a parámetros, pero la entrada se toma de un fichero en lugar del teclado.

### Función [fflush](http://c.conclase.net/librerias/?ansifun=fflush):

Sintaxis:

int fflush(FILE \*fichero);

Esta función fuerza la salida de los datos acumulados en el buffer de salida del *fichero*. Para mejorar las prestaciones del manejo de ficheros se utilizan buffers, almacenes temporales de datos en memoria, las operaciones de salida se hacen a través del buffer, y sólo cuando el buffer se llena se realiza la escritura en el disco y se vacía el buffer. En ocasiones nos hace falta vaciar ese buffer de un modo manual, para eso sirve ésta función.

El valor de retorno es cero si la función se ejecutó con éxito, y EOF si hubo algún error. El parámetro de entrada es un puntero a la estructura [FILE](http://c.conclase.net/librerias/?ansitip=FILE) del fichero del que se quiere vaciar el buffer. Si es [NULL](http://c.conclase.net/librerias/?ansimac=tNULL) se hará el vaciado de todos los ficheros abiertos.

Referencia Bibliográfica:

URL: <http://c.conclase.net/ficheros/?cap=002>