

File Write and Read

Yung-Hsiang Lu

fopen, fprintf, fclose

The image shows a code editor window with a file named `write.c` and a terminal window. The code in `write.c` demonstrates the use of `fopen`, `fprintf`, and `fclose`. Annotations highlight key parts of the code and the terminal output.

```
#include <stdio.h>
int main(int argc, char * argv[])
{
    FILE * fh = fopen("message", "w");
    if (fh == NULL)
    {
        printf("fopen fail\n");
        return -1; /* abnormal termination */
    }
    fprintf(fh, "This is a message.\n");
    fprintf(fh, "Another line.\n");
    fclose(fh);
    return 0;
}
```

Annotations in the code editor:

- file name** (yellow box) points to the filename `"message"` in the `fopen` call.
- write to the file** (green box) points to the `"w"` mode in the `fopen` call.
- print to the file** (cyan box) points to the `fprintf` calls.
- Close the file after finish writing** (red text) points to the `fclose(fh);` line.

The terminal window shows the output of the program:

```
[Ubuntu Linux ] more message
This is a message.
Another line.
[Ubuntu Linux ] █
```

fscanf

```
File Edit Navigate Project Window Help
read.c x
#include <stdio.h>
int main(int argc, char * argv[])
{
    FILE * fh = fopen("message", "r");
    if (fh == NULL)
    {
        printf("fopen fail\n");
        return -1; /* abnormal termination */
    }
    char line[80];
    while (! feof(fh))
    {
        fscanf(fh, "%s", line);
        printf("%s\n", line);
    }
    fclose(fh);
    return 0;
}
Console x
<terminated> Function (2) [C/
This
is
a
message.
Another
line.
line.
```

feof = end of file

Writable Smart Insert 19 :

manpage = manual page

NAME

`fscanf`, `scanf`, `sscanf`, `vfscanf`, `vscanf`, `vsscanf` -- input format conversion

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

```
#include <stdio.h>
```

```
int  
fscanf(FILE *restrict stream, const char *restrict format, ...);
```

```
int  
scanf(const char *restrict format, ...);
```

```
int  
sscanf(const char *restrict s, const char *restrict format, ...);
```

```
#include <stdarg.h>  
#include <stdio.h>
```

```
int  
vfscanf(FILE *restrict stream, const char *restrict format, va_list arg);
```

man fscanf(3) - Mozilla Firefox

File Edit View History Bookmarks Yahoo! Tools Help

http://www.manpagez.com/man/3/fscanf/

be a pointer to *unsigned int*.

x, X Matches an optionally signed hexadecimal integer; the next pointer must be a pointer to *unsigned int*.

a, A, e, E, f, F, g, G Matches a floating-point number in the style of **strtod(3)**. The next pointer must be a pointer to *float* (unless **l** or **L** is specified.)

s Matches a sequence of non-white-space characters; the next pointer must be a pointer to *char*, and the array must be large enough to accept all the sequence and the terminating NUL character. The input string stops at white space or at the maximum field width, whichever occurs first.

If an **l** qualifier is present, the next pointer must be a pointer to *wchar_t*, into which the input will be placed after conversion by **mbrtowc(3)**.

S The same as **ls**.

c Matches a sequence of *width* count characters (default 1); the next pointer must be a pointer to *char*, and there must be enough room for all the characters (no terminating NUL is added). The usual skip of leading white space is suppressed. To skip white space first, use an explicit space in the format.

If an **l** qualifier is present, the next pointer must be a pointer to *wchar_t*, into which the input will be placed after conversion by **mbrtowc(3)**.

C The same as **lc**.

Done

Yung-Hsiang Lu

5

man fopen(3) - Mozilla Firefox

File Edit View History Bookmarks Yahoo! Tools Help

http://www.manpagez.com/man/3/fopen/

DESCRIPTION

The **fopen()** function opens the file whose name is the string pointed to by *filename* and associates a stream with it.

The argument *mode* points to a string beginning with one of the following sequences (Additional characters may follow these sequences.):

- ```r``` Open text file for reading. The stream is positioned at the beginning of the file.
- ```r+``` Open for reading and writing. The stream is positioned at the beginning of the file.
- ```w``` Truncate file to zero length or create text file for writing. The stream is positioned at the beginning of the file.
- ```w+``` Open for reading and writing. The file is created if it does not exist, otherwise it is truncated. The stream is positioned at the beginning of the file.
- ```a``` Open for writing. The file is created if it does not exist. The stream is positioned at the end of the file. Subsequent writes to the file will always end up at the then current end of file, irrespective of any intervening **fseek(3)** or similar.
- ```a+``` Open for reading and writing. The file is created if it does not exist. The stream is positioned at the end of the file. Subsequent writes to the file will always end up at the then current end of file, irrespective of any intervening **fseek(3)** or similar.

Yung-Hsiang Lu

Done 6

fgets

The screenshot shows the Eclipse IDE with a C program named `read.c` open. The code uses `fopen` to open a file named `message` in read mode. It then uses `fgets` to read the file line by line into a character array `line`. The output of the program is shown in the console window, displaying two lines of text: `This is a message.` and `Another line.`

```
#include <stdio.h>
int main(int argc, char * argv[])
{
    FILE * fh = fopen("message", "r");
    if (fh == NULL)
    {
        printf("fopen fail\n");
        return -1; /* abnormal termination */
    }
    char line[81]; /* assume no line has more than 80 charac
    while (fgets(line, 80, fh) != NULL)
    {
        printf("%s\n", line);
    }
    fclose(fh);
    return 0;
}
```

Annotations in the image include:

- A yellow box with the text "read a whole line" and a yellow arrow pointing to the `while` loop in the code.
- A green box with the text "Add an additional new line" and a green arrow pointing to the `printf` statement in the code.

The console window on the right shows the output of the program:

```
This is a message.
Another line.
```

NAME

fgets, **gets** -- get a line from a stream

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

```
#include <stdio.h>
```

```
char *  
fgets(char *restrict s, int n, FILE *restrict stream);
```

```
char *  
gets(char *s);
```

DESCRIPTION

DESCRIPTION



The **fgets()** function reads at most one less than the number of characters specified by *n* from the given *stream* and stores them in the string *s*. Reading stops when a newline character is found, at end-of-file or error. The newline, if any, is retained. If any characters are read and there is no error, a `\0` character is appended to end the string.

The **gets()** function is equivalent to **fgets()** with an infinite *n* and a *stream* of `stdin`, except that the newline character (if any) is not stored in the string. It is the caller's responsibility to ensure that the input line, if any, is sufficiently short to fit in the string.

RETURN VALUES



Upon successful completion, **fgets()** and **gets()** return a pointer to the string. If end-of-file occurs before any characters are read, they return `NULL` and the buffer contents remain unchanged. If an error occurs, they return `NULL` and the buffer contents are indeterminate. The **fgets()** and **gets()** functions do not distinguish between end-of-file and error; callers must use **feof(3)** and **ferror(3)** to determine which occurred.

ERRORS

FILE * fh = fopen("filename", "r"); which is correct?

- A) If fh is NULL, fopen fails.
- B) The file is open for writes.
- C) If fh is zero, fopen succeeds.
- D) feof(fh) is false if the end of file has been reached.

Correct - Click anywhere to continue

Incorrect - Click anywhere to continue

Your answer:

You did not answer this question

You must answer the question before continuing

Submit

Clear

Close File

Which function is used to close an opened file?

Correct - Click anywhere to continue

Incorrect - Click anywhere to continue

Your answer:

You did not answer this question

You must answer the question before continuing

Submit

Clear

File

Your Score	{score}
Max Score	{max-score}
Number of Quiz Attempts	{total-attempts}

Question Feedback/Review Information Will Appear Here

Continue

Review Quiz