【Linux Programming】Day13

Tags	
Date	@June 5, 2022
Summary	C I/O Library Functions

【Ch3】Work with Files

3.5 The Standard I/O Library

In many ways, we use the library functions in the same way that we use low-level file descriptors.

The equivalent of the low-level file descriptor is called a stream and is implemented as a pointer to a structure, a `FILE *`.

Three file streams are automatically opened when a program is started. They are `stdin`, `stdout`, and `stderr`. All of these are declared in `stdio.h`.

3.5.1 fopen

```
#include <stdio.h>

FILE *fopen(const char *filename, const char *mode);
```

The `mode` parameter specifies how the file is to be opened. It's one of the following strings:

- "r" or "rb": Open for reading only
- "w" or "wb": Open for writing, truncate to zero length
- "a" or "ab": Open for writing, append to end of file
- "r+" or "rb+" or "r+b": Open for update(reading and writing)

- “w+” or “wb+” or “w+b”: Open for update, truncate to zero length
- “a+” or “ab+” or “a+b”: Open for update, append to end of file.

The `b` indicates that the file is a binary file rather than a text file.

Note: The mode parameter is always a string. Always use double quote instead of single quote.

If it fails, it returns the value `NULL`. The number of available streams is limited, `FOPEN_MAX` defined in `stdio.h`, which is usually 16 on Linux system.

3.5.2 fread

The `fread` library function is used to read data from a file stream. Both `fread` and `fwrite` deal with data records. These are specified by a record size, `size`, and a count, `nitems`, of records to transfer.

The function returns the number of items(rather than the number of bytes) successfully read into the data buffer.

```
#include <stdio.h>

size_t fred(void *ptr, size_t size, size_t nitems, FILE *stream);
```

3.5.3 fwrite

The fwrite library function takes data records from the specified data buffer and writes them to the output stream. It returns the number of records successfully written.

```
#include <stdio.h>

size_t fwrite(const void *ptr, size_t size, size_t nitems, FILE *stream);
```

3.5.4 fclose

The `fclose` library function closes the specified stream, causing any unwritten data to be written. It's important to use `fclose` because the `stdio` library will buffer data.

However, `fclose` is called automatically on all file streams that are still open when a program ends normally.

```
#include <stdio.h>

int fclose(FILE *stream);
```

3.5.5 fflush

The `fflush` library function causes all outstanding data on a file stream to be written immediately.

For example, we can use this to ensure that an interactive prompt has been sent before attempting to read a response.

```
#include <stdio.h>

int fflush(FILE *stream);
```

3.5.6 fseek

The `fseek` function is the file stream equivalent of the `lseek` system call. However, `fseek` returns an integer: 0 if it succeeds, -1 if it fails.

```
#inlcude <stdio.h>

int fseek(FILE *stream, long int offset,  int whence);
```

3.5.7 fgets, getc, and getchar

The `fetc` function returns the next byte as a character from a file stream.

When it reaches the file or there is an error, it returns `EOF`. We must use `ferror` of `feof` to distinguish the two cases.

```
#include <stdio.h>

int getc(FILE *stream);
int fgetc(FILE *stream);
int getchar();
```

The `getc` function is equivalent to `getc`, except that it may be implemented as a macro.

The `getchar` function is equivalent to `getc(stdin)`.

3.5.8 fputc, putc, and putchar

The `fputc` function writes a character to an output file stream. It returns the value it has written, or `EOF` on failure.

```
#include <stdio.h>

int fputc(int c, FILE *stream);
int putc(int c, FILE *stream);
int putchar(int c);
```

The `putchar` function is equivalent to `putc(c, stdout)`

3.5.9 fgets and gets

The `fgets` function reads a string from an input file stream.

```
#include <stdio.h>

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

`fgets` writes characters to the string pointed to by s until a newline is encountered, n-1characters have been transferred, or the ned of file is reached.