File (1A)

Young Won Lim
11/25/16

Copyright (c) 2010-2016 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

FILE Pointer and Functions

```
FILE *fp;
```

fopen	opens a file
fprintf	prints formatted wide character to a file stream
fscanf	read formatted input from a file stream
fclose	closes a file

https://en.wikipedia.org/wiki/C_file_input/output

Text and Binary File

Formatted Input / Output (Text Mode)

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

Unformatted Input / Output (Text Mode)

```
int fputc (int c, FILE *stream);
int fgetc (FILE *stream);
```

Binary Stream Input / Output (Binary Mode)

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

https://en.wikipedia.org/wiki/C_file_input/output

Formatted Input / Output (Text Mode)

```
int scanf    ( const char *format, ... );
int fscanf   ( FILE *stream, const char *format, ... );
int sscanf   ( const char *buffer, const char *format, ... );

int printf   ( const char *format, ... );
int fprintf  ( FILE *stream, const char *format, ... );
int sprintf  ( char *buffer, const char *format, ... );
```

https://en.wikipedia.org/wiki/C_file_input/output

Unformatted Input / Output (Text Mode)

```
int     fgetc    (FILE *stream);
char *fgets      (char *s, int size, FILE *stream);
int     getc     (FILE *stream);
int     getchar  (void);
~~char *gets       (char *s);~~
int     ungetc   (int c, FILE *stream);

int     fputc    (int c, FILE *stream);
int     fputs    (const char *s, FILE *stream);
int     putc     (int c, FILE *stream);
int     putchar  (int c);
int     puts     (const char *s);
```

https://en.wikipedia.org/wiki/C_file_input/output

Direct Input / Output

```
size_t fread  (void *buffer,        // where the read objects are stored
               size_t size,         // size of each object in bytes
               size_t count,        // the number of the objects
               FILE *stream);       // the stream to read

size_t fwrite (const void *buffer,  // where the objects are written
               size_t size,         // size of each object in bytes
               size_t count,        // the number of the objects
               FILE *stream);       // the stream to read
```

https://en.wikipedia.org/wiki/C_file_input/output

fopen()

FILE ***fopen**(const char *path, const char *mode);

opens a file and associates a stream with it
The file name is the string pointed to by path
mode points to a string consists of the following characters

Returns a FILE pointer (successful)
Returns NULL pointer and set errno (unsuccessful)

https://en.wikipedia.org/wiki/C_file_input/output

Mode

r	for reading	fpos_t : beginning
r+	for reading and writing	fpos_t : beginning
w	for writing	fpos_t : beginning
w+	for reading and writing	fpos_t : beginning
a	for appending	fpos_t : end
a+	for reading and appending	fpos_t : end(append), begining(read)

https://en.wikipedia.org/wiki/C_file_input/output

File Positioning

long **ftell**(FILE *stream);

returns the current file position indicator

int **fgetpos**(FILE *stream, fpos_t *pos);

gets the file position indicator

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

moves the file position indicator to a specific location in a file

int **fsetpos**(FILE *stream, const fpos_t *pos);

moves the file position indicator to a specific location in a file

void **rewind**(FILE *stream);

removes the file position indicator to the beginning in a file

https://en.wikipedia.org/wiki/C_file_input/output

Error Handling

void **clearerr**(FILE *stream);
clears the end-of-file and error indicators

int **feof**(FILE *stream);
tests the end-of-file indicator
returning nonzero if it is set.

int **ferror**(FILE *stream);
tests the end-of-file indicator
returning nonzero if it is set.

https://en.wikipedia.org/wiki/C_file_input/output

FILE Structure

FILE :

- known as a file handle
- an opaque type
- containing the *information* about a file or text stream needed to perform *input* or *output* operations on it,

an **opaque pointer** is a special case of an opaque data type, a datatype declared to be a pointer to a record or data structure of **some unspecified type**.

https://en.wikipedia.org/wiki/C_file_input/output

FILE Structure

containing the information about a file or text stream

- platform-specific identifier of the associated I/O device,
  such as a *file descriptor*
- the *buffer*
- *stream orientation* indicator (unset, narrow, or wide)
- *stream buffering* state indicator (unbuffered, line buffered, fully buffered)
- *I/O mode* indicator (input stream, output stream, or update stream)
- *binary/text mode* indicator
- *end-of-file* indicator
- *error* indicator
- the *current stream position* and
- *multibyte conversion* state (an object of type fpos_t)
- reentrant *lock* (required as of C11)

https://en.wikipedia.org/wiki/C_file_input/output

FILE Structure

fpos_t –
a non-array type
capable of uniquely identifying the position of every byte in a file and
every conversion state that can occur in all supported multibyte character encodings

size_t –
an unsigned integer type
which is the type of the result of the sizeof operator.

https://en.wikipedia.org/wiki/C_file_input/output

References

[1] Essential C, Nick Parlante
[2] Efficient C Programming, Mark A. Weiss
[3] C A Reference Manual, Samuel P. Harbison & Guy L. Steele Jr.
[4] C Language Express, I. K. Chun