Block I/O: fread( ) and fwrite( )

1 Overview

Standard I/O library:

fopen( ), fclose( )	- opening and closing files
fprintf( ), fscanf( )	- field at a time with data conversion
fgetc( ), fputc( )	- character (byte) at a time
fgets( ), fputs( )	- line at a time
fread( ), fwrite( ), fseek( )	- physical block at a time

The **fread( )** and **fwrite( )** functions are the most efficient way to read or write large amounts of data.

2 fread( )

The **fread( )** function reads a specified number of bytes from a binary file and places them into memory at the specified location.

Prototype:

```
int fread(void *InArea, int elementSize, int count, FILE *fp);
```

`int` return type: the function returns the number of items read
`void *InArea`: a pointer to the memory area to be read into
`int elementSize`: size of the basic data item, often specified using the `sizeof` operator
`int count`: number of data items
`FILE *fp`: the pointer to the file that was returned by `fopen( )`

Example:

```
typedef struct pixel_type {
    unsigned char r;
    unsigned char g;
    unsigned char b;
} pixel_t;

typedef struct image_type {
    int height;
    int width;
    pixel_t *pixels;
} image_t;

void parseHeader(FILE *inFilePtr, image_t *theImage);
void writeHeader(FILE *outFilePtr, image_t *theImage);
void writeImage(image_t *theImage);

int main(void) {
    image_t *image;
    FILE *in;
    int howMany = 0;

    in = fopen("INfilename", "r");
    if (in == NULL) {
        fprintf(stderr, "Couldn't open file for reading. \n");
        exit(1);
    }
    parseHeader(in, image);

    image->pixels = (pixel_t *) malloc(sizeof(pixel_t) * image->height * image->width);
    howMany = fread(image->pixels, sizeof(pixel_t), image->height * image->width, in);

    if (howMany != image->height * image->width) {
        fprintf(stderr, "Read error, wanted %d got %d \n", image->height * image->width, howMany);
        exit(1);
    }

    writeImage(image);    <--- this function next page in the fwrite( ) example
    ... rest of program
```

3 fwrite( )

The **fwrite( )** function writes a specified number of bytes from the memory address specified and places them into the file.

Prototype:

```
int fwrite(void *OutArea, int elementSize, int count, FILE *fp);
```

`int` return type: the function returns the total number of characters written
`void *OutArea`: a pointer to the memory area holding the data to be written
`int elementSize`: size of the basic data item, often specified using the `sizeof` operator
`int count`: number of data items
`FILE *fp`: the pointer to the file that was returned by `fopen( )`

Example (a continuation from the `fread( )` example previous page):

```
void writeImage(image_t *theImage) {
    int row = 0;
    int howMany = 0;
    FILE *out;

    out = fopen("OUTfilename", "w");
    if (out == NULL) {
        fprintf(stderr, "Couldn't open file for writing. \n");
        exit(1);
    }

    writeHeader(out, theImage);

    howMany = fwrite(theImage->pixels, sizeof(pixel_t), theImage->height * theImage->width, out);

    if (howMany != theImage->height * theImage->width) {
        fprintf(stderr, "Write error %d \n", howMany);
        exit(1);
    }

    fclose(out);
}
```