

Знакомство с файловой системой



Файлы

Файл - это именованный блок информации, расположенный на носителе информации.

Любой файл обладает следующим рядом особенностей:

- Файл не может располагаться на диске непрерывно, однако пользователю файл предоставляется цельным блоком последовательной байтовой информации.
- Название файла не может содержать символы: < > : " / \ |.
- Файл обладает расширением - сочетанием символов, с помощью которых операционная система определяет тип файла. Расширение - необязательная часть.
- У каждого файла есть, так называемые атрибуты, которые, например, определяют уровни доступа к нему. Используя атрибуты, операционная система узнает, как нужно и, главное, можно, работать с данным файлом.

Открытие файла

Прототип функции fopen:

```
FILE * fopen_s(FILE *file, const char * fname, const char * modeopen);
```

Функция fopen_s открывает файл, имя которого указано в параметре fname и связывает его с потоком, который может быть идентифицирован для выполнения различных операций с файлом.

Операции с потоком, выполнение которых разрешено определяются параметром modeopen.

Открытие файла

Указатель файла - это указатель на структуру типа FILE. В программе, прежде всего, следует задать указатель на структуру FILE:

```
FILE *fp;
```

Режим открытия	Описание
r	Файл открывается только для чтения.
w	Файл открывается только для записи. Если файл не существует, то он создается. Если же файл существует, то он будет переписан.
a	Файл открывается только для дозаписи (начиная с конца файла). Файл открывается для записи данных в конец или создается для записи, если он не существует.
r+	Существующий файл открывается для обновления (чтения и записи).
w+	Создается новый файл для обновления (чтения и записи). Если файл существует, то он будет перезаписан.
a+	Файл открывается для добавления, т.е. записи с конца файла. Если файл не существует, то он создается.

Дескриптор файла и файловый указатель

1. **Дескриптор файла** - уникальный номер, который операционная система присваивает любому открытому файлу, чтобы отличать его от других. Когда файл закрывается, система "отбирает" у него дескриптор. Именно это уникальное число мы будем использовать для работы с конкретным файлом в наших программах.

2. **Файловый указатель** - специальная переменная, которая автоматически присваивается открытому файлу и хранит текущую позицию в файле. Она перемещается по файлу в момент процессов записи и чтения. Для большего понимания, вы можете представить данную переменную в виде курсора в любом текстовом редакторе.

Разновидности файлов

Текстовый файл	Двоичный файл
<p>Число, записанное в файл — запишется как текст, то есть размер занятого пространства будет равен количеству цифр в этом числе.</p>	<p>Число, записанное в файл — запишется в двоичном формате, и размер занятого пространства будет равен размеру типа данных в этом числе.</p>
<p>Прочитать данный файл можно с помощью текстового редактора.</p>	<p>Прочитать файл можно только с помощью особой программы.</p>
<p>При потере нескольких байт, информацию можно восстановить по смыслу.</p>	<p>Потеря нескольких байт в двоичном файле может быть необратима</p>
<p>Считывание информации программным путем затруднено, так как файл представляет собой единый текстовый блок.</p>	<p>Считывание информации программным путем облегчено, так как файл представляет собой набор блоков информации с конкретными типами данных.</p>
<p>Используется как файл для чтения, редактирования, вывода на печать.</p>	<p>Используется как файл для удобного чтения, хранения, записи информации программным путем.</p>

Функции библиотеки cstdio

Функция	Пояснение
<pre> errno_t fopen_s (FILE **file, const char *filename, const char *mode) </pre>	<p>функция открывает файл.</p> <p>file - указатель на файловый указатель открытого файла.</p> <p>filename - путь к файлу</p> <p>mode - тип доступа:</p> <ul style="list-style-type: none"> •r - чтение, если файла нет, то данная функция генерирует ошибку (возвращает 0) •w - запись, если файла нет, то файл создаётся, если есть исходное содержимое удаляется •a - добавление в конец, если файла нет, то он создаётся •r+ чтение и запись (файл должен существовать) •w+ - чтение и запись (принцип работы как у w) •a+ - добавление и чтение (принцип работы как у a)
<pre> int fclose (FILE *stream) </pre>	<p>функция закрывает файл</p> <p>stream - указатель на закрываемый файл.</p>

Пример работы

```
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
    FILE * pFile;
    fopen_s(&pFile, "file.txt", "w");
    if (pFile != NULL)
    {
        fputs("Я выучу C++!!!. ", pFile);
        // записать строку в файл
        fclose(pFile);
    }
    else
    {
        cout << "ERROR" << endl;
    }
    system("pause");
    return 0;
}
```


Пример работы

```
#include <cstdio>
#include <iostream>
using namespace std;
int main()
{
    setlocale(LC_ALL, "rus");
    FILE * ptrFile;
    fopen_s(&ptrFile, "d:\\file.txt", "r");
    char mystring[100];
    if (ptrFile == NULL)
        cout << "ERROR";
    else
    {
        if (fgets(mystring, 100, ptrFile) != NULL)
            puts(mystring);
        fclose(ptrFile);
    }
    system("pause");
    return 0;
}
```

Функции библиотеки `cstdio`

Функция	Пояснение
<code>char *fgets (char *string, int n, FILE *stream)</code>	считывает строку начиная с текущей позиции. <u>string</u> - строка, в которую попадают считанные данные. n - количество элементов в <u>string</u> . <u>stream</u> - указатель на открытый файл.
<code>int fputs (const char *string, FILE *stream)</code>	записывает строку в файл, начиная с текущей позиции. <u>string</u> - строка для записи. <u>stream</u> - указатель на открытый файл, куда производится запись.
<code>size_t fwrite (const void *buffer, size_t size, size_t count, FILE *stream)</code>	функция записывает массив данных в файл. <u>buffer</u> - адрес массива, где содержатся данные <u>size</u> - размер элемента массива в байтах <u>count</u> - максимальное количество элементов для записи в файл <u>stream</u> - указатель на открытый файл.

Функции библиотеки `cstdio`

Функция	Пояснение
<code>size_t fread(void *buffer, size_t size, size_t count, FILE *stream)</code>	<p>функция считывает данные из файла в буфер.</p> <p><code>buffer</code> - адрес массива, куда запишутся данные.</p> <p><code>size</code> - размер элемента массива в байтах.</p> <p><code>count</code> - максимальное количество элементов для считывания.</p> <p><code>stream</code> - указатель на открытый файл.</p>
<code>int feof (FILE *stream)</code>	функция проверяет достигнут ли конец файла.

Функции библиотеки cstdio

```
#include <cstdio>
#include <iostream>
void main()
{
    setlocale(0, "");
    FILE *readFile, *writeFile;
    // открытие файла для чтения
    if (fopen_s(&readFile, "file.txt", "r") != NULL)
        printf("Ошибка открытия файла");
    else
    {
        // открытие файла для записи
        if (fopen_s(&writeFile, "file1.txt", "w") != NULL)
            printf("Ошибка открытия файла");
        else
        {
            char mystring[100];
            // считать символы из файла
            if (fgets(mystring, 100, readFile) != nullptr)
                fputs(mystring, writeFile); // запись в файл
            fclose(readFile);
            fclose(writeFile);
        }
    }
    system("pause");
}
```

<http://cppstudio.com/cat/309/323/>