

🏠 (<http://cppstudio.com>) / Язык программирования C++ (<http://cppstudio.com/cat/274/>)
/ Структуры и файлы (<http://cppstudio.com/cat/274/279/>) / Работа с файлами в C++

Работа с файлами в C++



Оценка: **4,53** (голосов: 17)
Чтобы проголосовать, вы должны зарегистрироваться.

Большинство компьютерных программ работают с файлами, и поэтому возникает необходимость создавать, удалять, записывать, читать, открывать файлы. Что же такое файл? Файл – именованный набор байтов, который может быть сохранен на некотором накопителе. Ну, теперь ясно, что под файлом понимается некоторая последовательность байтов, которая имеет своё, уникальное имя, например файл.txt. В одной директории не могут находиться файлы с одинаковыми именами. Под именем файла понимается не только его название, но и расширение, например: file.txt и file.dat – разные файлы, хоть и имеют одинаковые названия. Существует такое понятие, как полное имя файлов – это полный адрес к директории файла с указанием имени файла, например: D:\docs\file.txt. Важно понимать эти базовые понятия, иначе сложно будет работать с файлами.

Для работы с файлами необходимо подключить заголовочный файл `<fstream>`. В `<fstream>` определены несколько классов и подключены заголовочные файлы `<ifstream>` – файловый ввод и `<ofstream>` – файловый вывод.

Файловый ввод/вывод аналогичен стандартному вводу/выводу, единственное отличие – это то, что ввод/вывод выполняется не на экран, а в файл. Если ввод/вывод на стандартные устройства выполняется с помощью объектов `cin` и `cout`, то для организации файлового ввода/вывода достаточно создать собственные объекты, которые можно использовать аналогично операторам `cin` и `cout`.

Например, необходимо создать текстовый файл и записать в него строку Работа с файлами в C++. Для этого необходимо проделать следующие шаги:

1. создать объект класса `ofstream`;
2. связать объект класса с файлом, в который будет производиться запись;
3. записать строку в файл;
4. закрыть файл.

Почему необходимо создавать объект класса `ofstream`, а не класса `ifstream`? Потому, что нужно сделать запись в файл, а если бы нужно было считать данные из файла, то создавался бы объект класса `ifstream`.

```
1 // создаём объект для записи в файл
2 ofstream /*имя объекта*/; // объект класса ofstream
```

Назовём объект – `fout`, Вот что получится:

```
1 ofstream fout;
```

Для чего нам объект? Объект необходим, чтобы можно было выполнять запись в файл. Уже объект создан, но не связан с файлом, в который нужно записать строку.

```
1 fout.open("cppstudio.txt"); // связываем объект с файлом
```

Через операцию точка получаем доступ к методу класса `open()`, в круглых скобках которого указываем имя файла. Указанный файл будет создан в текущей директории с программой. Если файл с таким именем существует, то существующий файл будет заменен новым. Итак, файл открыт, осталось записать в него нужную строку. Делается это так:

```
1 fout << "Работа с файлами в C++"; // запись строки в файл
```

Используя операцию передачи в поток совместно с объектом `fout` строка `Работа с файлами в C++` записывается в файл. Так как больше нет необходимости изменять содержимое файла, его нужно закрыть, то есть отделить объект от файла.

```
1 fout.close(); // закрываем файл
```

Итог – создан файл со строкой `Работа с файлами в C++`.

Шаги 1 и 2 можно объединить, то есть в одной строке создать объект и связать его с файлом. Делается это так:

```
1 ofstream fout("cppstudio.txt"); // создаём объект класса ofstream и связываем его
```

Объединим весь код и получим следующую программу.

```
1 // file.cpp: определяет точку входа для консольного приложения.
2
3 #include "stdafx.h"
4 #include <fstream>
5 using namespace std;
6
7 int main(int argc, char* argv[])
8 {
9     ofstream fout("cppstudio.txt"); // создаём объект класса ofstream для записи
10    fout << "Работа с файлами в C++"; // запись строки в файл
11    fout.close(); // закрываем файл
12    system("pause");
13    return 0;
14 }
```

Осталось проверить правильность работы программы, а для этого открываем файл `cppstudio.txt` и смотрим его содержимое, должно быть – `Работа с файлами в C++`.

Для того чтобы прочитать файл понадобится выполнить те же шаги, что и при записи в файл с небольшими изменениями:

1. создать объект класса `ifstream` и связать его с файлом, из которого будет производиться считывание;
2. прочитать файл;
3. закрыть файл.

```
1 // file_read.cpp: определяет точку входа для консольного приложения.
2
3 #include "stdafx.h"
4 #include <fstream>
5 #include <iostream>
6 using namespace std;
7
8 int main(int argc, char* argv[])
9 {
10     setlocale(LC_ALL, "rus"); // корректное отображение Кириллицы
11     char buff[50]; // буфер промежуточного хранения считываемого из файла текста
12     ifstream fin("cppstudio.txt"); // открыли файл для чтения
13
14     fin >> buff; // считали первое слово из файла
15     cout << buff << endl; // напечатали это слово
16
17     fin.getline(buff, 50); // считали строку из файла
18     fin.close(); // закрываем файл
19     cout << buff << endl; // напечатали эту строку
20
21     system("pause");
22     return 0;
23 }
```

В программе показаны два способа чтения из файла, первый – используя операцию передачи в поток, второй – используя функцию `getline()`. В первом случае считывается только первое слово, а во втором случае считывается строка, длиной 50 символов. Но так как в файле осталось меньше 50 символов, то считываются символы включительно до последнего. Обратите внимание на то, что считывание во второй раз (строка 17) продолжилось, после первого слова, а не с начала, так как первое слово было прочитано в строке 14. Результат работы программы показан на рисунке 1.

CppStudio.com

Работа

с файлами в C++

Для продолжения нажмите любую клавишу . . .

Рисунок 1 – Работа с файлами в C++

Программа сработала правильно, но не всегда так бывает, даже в том случае, если с кодом всё в порядке. Например, в программу передано имя несуществующего файла или в имени допущена ошибка. Что тогда? В этом случае ничего не произойдёт вообще. Файл не будет найден, а значит и прочитать его не возможно. Поэтому компилятор проигнорирует строки, где выполняется работа с файлом. В результате корректно завершится работа программы, но ничего, на экране показано не будет. Казалось бы это вполне нормальная реакции на такую ситуацию. Но простому пользователю не будет понятно, в чём дело и почему на экране не появилась строка из файла. Так вот, чтобы всё было предельно понятно в C++ предусмотрена такая функция – `is_open()`, которая возвращает целые значения: 1 – если файл был успешно открыт, 0 – если файл открыт не был. Доработаем программу с открытием файла, таким образом, что если файл не открыт выводилось соответствующее сообщение.

```
1 // file_read.cpp: определяет точку входа для консольного приложения.
2
3 #include "stdafx.h"
4 #include <fstream>
5 #include <iostream>
6 using namespace std;
7
8 int main(int argc, char* argv[])
9 {
10     setlocale(LC_ALL, "rus"); // корректное отображение Кириллицы
11     char buff[50]; // буфер промежуточного хранения считываемого из файла текста
12     ifstream fin("cppstudio.doc"); // (ВВЕЛИ НЕ КОРРЕКТНОЕ ИМЯ ФАЙЛА)
13
14     if (!fin.is_open()) // если файл не открыт
15         cout << "Файл не может быть открыт!\n"; // сообщить об этом
16     else
17     {
18         fin >> buff; // считали первое слово из файла
19         cout << buff << endl; // напечатали это слово
20
21         fin.getline(buff, 50); // считали строку из файла
22         fin.close(); // закрываем файл
23         cout << buff << endl; // напечатали эту строку
24     }
25     system("pause");
26     return 0;
27 }
```

Результат работы программы показан на рисунке 2.

CppStudio.com

Файл не может быть открыт!

Для продолжения нажмите любую клавишу . . .

Рисунок 2 – Работа с файлами в C++

Как видно из рисунка 2 программа сообщила о невозможности открыть файл. Поэтому, если программа работает с файлами, рекомендуется использовать эту функцию, `is_open()`, даже, если уверены, что файл существует.

Режимы открытия файлов

Режимы открытия файлов устанавливают характер использования файлов. Для установки режима в классе `ios_base` предусмотрены константы, которые определяют режим открытия файлов (см. Таблица 1).

Таблица 1 – режимы открытия файлов

Константа	Описание
<code>ios_base::in</code>	открыть файл для чтения
<code>ios_base::out</code>	открыть файл для записи
<code>ios_base::ate</code>	при открытии переместить указатель в конец файла
<code>ios_base::app</code>	открыть файл для записи в конец файла

Константа	Описание
<code>ios_base::trunc</code>	удалить содержимое файла, если он существует
<code>ios_base::binary</code>	открытие файла в двоичном режиме

Режимы открытия файлов можно устанавливать непосредственно при создании объекта или при вызове функции `open()`.

```
1 ofstream fout("cppstudio.txt", ios_base::app); // открываем файл для добавления
2 fout.open("cppstudio.txt", ios_base::app); // открываем файл для добавления инфо
```

Режимы открытия файлов можно комбинировать с помощью поразрядной логической операции **или** `|`, например: `ios_base::out | ios_base::trunc` – открытие файла для записи, предварительно очистив его.

Объекты класса `ofstream`, при связке с файлами по умолчанию содержат режимы открытия файлов `ios_base::out | ios_base::trunc`. То есть файл будет создан, если не существует. Если же файл существует, то его содержимое будет удалено, а сам файл будет готов к записи. Объекты класса `ifstream` связываясь с файлом, имеют по умолчанию режим открытия файла `ios_base::in` – файл открыт только для чтения. Режим открытия файла ещё называют – флаг, для удобочитаемости в дальнейшем будем использовать именно этот термин. В таблице 1 перечислены далеко не все флаги, но для начала этих должно хватить.

Обратите внимание на то, что флаги `ate` и `app` по описанию очень похожи, они оба перемещают указатель в конец файла, но флаг `app` позволяет производить запись, только в конец файла, а флаг `ate` просто переставляет флаг в конец файла и не ограничивает места записи.

Разработаем программу, которая, используя операцию `sizeof()`, будет вычислять характеристики основных типов данных в C++ и записывать их в файл. Характеристики:

1. число байт, отводимое под тип данных
2. максимальное значение, которое может хранить определённый тип данных.

Запись в файл должна выполняться в таком формате:

```
1 /* data type      byte      max value
2 bool              = 1       255.00
3 char              = 1       255.00
4 short int         = 2       32767.00
5 unsigned short int = 2       65535.00
6 int               = 4       2147483647.00
7 unsigned int      = 4       4294967295.00
8 long int          = 4       2147483647.00
9 unsigned long int = 4       4294967295.00
10 float            = 4       2147483647.00
11 long float       = 8       9223372036854775800.00
12 double           = 8       9223372036854775800.00 */
```

Такая программа уже разрабатывалась ранее в разделе Типы данных C++ (http://cppstudio.com/obuchenie_cpp/typy_dannyh), но там вся информация о типах данных выводилась на стандартное устройство вывода, а нам необходимо программу переделать так, чтобы информация записывалась в файл. Для этого необходимо открыть файл в режиме записи, с предварительным усечением текущей

информации файла (строка 14). Как только файл создан и успешно открыт (строки 16 – 20), вместо оператора `cout`, в строке 22 используем объект `fout`. таким образом, вместо экрана информация о типах данных запишется в файл.

```

1  // write_file.cpp: определяет точку входа для консольного приложения.
2
3  #include "stdafx.h"
4  #include <iostream>
5  #include <fstream> // работа с файлами
6  #include <iomanip> // манипуляторы ввода/вывода
7  using namespace std;
8
9  int main(int argc, char* argv[])
10 {
11     setlocale(LC_ALL, "rus");
12
13     // связываем объект с файлом, при этом файл открываем в режиме записи, пред
14     ofstream fout("data_types.txt", ios_base::out | ios_base::trunc);
15
16     if (!fout.is_open()) // если файл небыл открыт
17     {
18         cout << "Файл не может быть открыт или создан\n"; // напечатать соответств
19         return 1; // выполнить выход из программы
20     }
21
22     fout << "    data type    " << "byte" << "
23         << "bool    = " << sizeof(bool) << "
24     /*вычисляем максимальное значение для типа данных bool*/
25     << "char    = " << sizeof(char) << "
26     /*вычисляем максимальное значение для типа данных char*/
27     << "short int    = " << sizeof(short int) << "
28     /*вычисляем максимальное значение для типа данных short int*/
29     << "unsigned short int = " << sizeof(unsigned short int) << "
30     /*вычисляем максимальное значение для типа данных unsigned short int*/
31     << "int    = " << sizeof(int) << "
32     /*вычисляем максимальное значение для типа данных int*/
33     << "unsigned int    = " << sizeof(unsigned int) << "
34     /*вычисляем максимальное значение для типа данных unsigned int*/
35     << "long int    = " << sizeof(long int) << "
36     /*вычисляем максимальное значение для типа данных long int*/
37     << "unsigned long int = " << sizeof(unsigned long int) << "
38     /*вычисляем максимальное значение для типа данных undigned long int*/
39     << "float    = " << sizeof(float) << "
40     /*вычисляем максимальное значение для типа данных float*/
41     << "long float    = " << sizeof(long float) << "
42     /*вычисляем максимальное значение для типа данных long float*/
43     << "double    = " << sizeof(double) << "
44     /*вычисляем максимальное значение для типа данных double*/
45     fout.close(); // программа больше не использует файл, поэтому его нужно
46     cout << "Данные успешно записаны в файл data_types.txt\n";
47     system("pause");
48     return 0;
49 }

```

Нельзя не заметить, что изменения в программе минимальны, а всё благодаря тому, что стандартный ввод/вывод и файловый ввод/вывод используются абсолютно аналогично. В конце программы, в строке 45 мы явно закрыли файл, хотя это и не обязательно, но считается хорошим тоном программирования. Стоит отметить, что все функции и манипуляторы используемые для форматирования стандартного ввода/вывода актуальны и для файлового ввода/вывода. Поэтому не возникло никаких ошибок, когда оператор `cout` был заменён объектом `fout`.

Практика



Обсудить на форуме (/topics/)

Автор: admin (/forums/users/admin/)

Дата: 25.08.2012

Поделиться:

←Перечисления в C++ (enum) (<http://cppstudio.com/post/8106/>)

Функции в C++ (<http://cppstudio.com/post/396/>)

Похожие статьи:

1. Типы данных C++ (<http://cppstudio.com/post/271/>)
2. Рекурсия в C++ (<http://cppstudio.com/post/418/>)
3. Слова сформированные из первых букв слов каждой строки файла (<http://cppstudio.com/post/1398/>)
4. Баланс фигурных скобок в файле (<http://cppstudio.com/post/1412/>)
5. Классы в C++ (<http://cppstudio.com/post/439/>)

Комментарии



Alex Nikolayuk

02.03.2018 (/post/446/comment-page-3/#comment-4056)

Доброго дня! Допоможіть будь ласка. Як можна зчитувати з файлу якщо не знаєш скільки в ньому символів? Наприклад коли зчитуєш числа, що розташовані через пробіл і потім їх потрібно записати в масив.

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)



varex83

(https://www.facebook.com/app_scoped_user_id/876212199217)

06.03.2018 (/post/446/comment-page-3/#comment-4060)

```
while (in>>x)
```

```
{cout<<x<<» «;
```

```
}
```

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)



Vova Denys

(<https://plus.google.com/10677063449931178050>)

14.03.2018 (/post/446/comment-page-3/#comment-4067)

```
1 ifstream file("cpp.txt");
2 while (!file.eof()) {
3
4 }
```

**npavelFax**

14.02.2017 (/post/446/comment-page-3/#comment-3487)

Заработок на дому официальная работа.

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)

**Денис Сидоренко** (<http://vk.com/id301100784>)

05.12.2016 (/post/446/comment-page-3/#comment-3357)

Подскажите пожалуйста. в папке с проектом есть, дочерняя пака initdata, которая в свою очередь имеет файл init.txt.

Можно ли как то указать путь к этому файлу без указания полного пути, например «initdata\init.txt», а не «C:\project\initdata\init.txt»?

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)

**Team_on**(https://www.facebook.com/app_scoped_user_id/214546252388)

21.10.2017 (/post/446/comment-page-3/#comment-3821)

```
1 unsigned long long FILE_LENGTH = 0;
2 for (FILE_LENGTH = 0; fin.peek() != EOF; FILE_LENGTH++)
3     fin.get();
4 }
5 cout << FILE_LENGTH << endl;
6 fin.seekg(0);
7 char *text = new char[FILE_LENGTH];
8 for (unsigned long long i = 0; fin.peek() != EOF; i++)
9     text[i] = fin.get();
10 }
11 text[FILE_LENGTH] = '\0';
12 //cout << text << endl;
```

```

1 unsigned long long FILE_LENGTH = 0;
2 for (FILE_LENGTH = 0; fin.peek() != EOF; FILE_LEN
3     fin.get());
4 }
5 cout << FILE_LENGTH << endl;
6 fin.seekg(0);
7 char *text = new char[FILE_LENGTH];
8 for (unsigned long long i = 0; fin.peek() != EOF;
9     text[i] = fin.get());
10 }
11 text[FILE_LENGTH] = '\0';
12 //cout << text << endl;

```

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)



gro

11.04.2015 (/post/446/comment-page-3/#comment-2634)

У меня в Дебиане следующая картина:

	data type	byte	max value
1	bool	1	255.00
2	char	1	255.00
3	sh int	2	65535.00
4	un sh int	2	65535.00
5	int	4	4294967295.00
6	uns int	4	4294967295.00
7	lo int	8	18446744073709551616.00
8	un lo int	8	18446744073709551616.00
9	float	4	4294967295.00
10	double	8	18446744073709551616.00
11	lo doub	16	340282366920938463463374607431768211456.00

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)



petruska

19.10.2014 (/post/446/comment-page-3/#comment-2161)

Помогите, когда считываю файл в строку или символьный массив, строка(массив) записует до первого пробела! Как записать в строку(массив) текст с пробелами?

Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)



Александр Михайлов (<http://vk.com/id378424>)

24.11.2015 (/post/446/comment-page-3/#comment-2987)

```
1 .getline(buff, 50)
```




Войдите, чтобы ответить (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F)

← Старые комментарии (<http://cppstudio.com/post/446/comment-page-2/#comments>)

Оставить комментарий

Вы должны войти (http://cppstudio.com/wp-login.php?redirect_to=http%3A%2F%2Fcppstudio.com%2Fpost%2F446%2F), чтобы оставить комментарий.

Translation

 (/post/446/)Русский (/post/446/)
 (/uk/post/446/)Українська (/uk/post/446/)
 (/en/post/446/)English (/en/post/446/)
 (/de/post/446/)Deutsch (/de/post/446/)
 (/be/post/446/)Беларуская (/be/post/446/)
 (/kk/post/446/)Қазақ тілі (/kk/post/446/)
 (/uz/post/446/)O'zbek tili (/uz/post/446/)
 (/tr/post/446/)Türkçe (/tr/post/446/)

Новое

- Особенности Qt: слоты и сигналы, описание QObject и QApplication, виды окон и т.д.
(<http://cppstudio.com/post/11167/>)
- Первая программа на Qt:
(<http://cppstudio.com/post/11127/>)
- Введение – графическая библиотека Qt
(<http://cppstudio.com/post/11097/>)
- Наследование классов
(<http://cppstudio.com/post/10103/>)
- Перегрузка операторов в C++ (часть 2)
(<http://cppstudio.com/post/10058/>)

Популярное

Sorry. No data so far.

© 2022 CppStudio - Программирование для начинающих на C++



(<https://plus.google.com/u/0/106109650739084338784>)

(<http://www.liveinternet.ru/click>)



(<http://orphus.ru>)