Read / write File in C ++

So far, we have used the iostream standard library, provided cin and cout methods to read from Standard Input and write to the corresponding Standard Output.

So far, we have used the **iostream** standard library, provided **cin and cout** methods to read from Standard Input and write to the corresponding Standard Output.

This chapter will show you how to read and write a file. This requires another C ++ Standard Library, **fstream** , which defines three new data types:

Data type Descriptionofstream This data type represents Output File Stream and is used to create files and to record information to those files ifstream This data type represents Input File Stream and is used to read information from the fstream file This data type generally represents File Stream, and has the capabilities of both ofstream and ifstream, meaning it can create files, write information to files and read information from files
To perform file processing in C ++, you include header files and in the source file of your C ++ program.

Open a File in C ++

A file must be opened before you can read information from it or write information to it. Or the **ofstream** object or **fstream** object can be used to open a file with the purpose of writing or ifstream object is used to open the file for reading purposes only.

Here is the standard syntax for open () function, which is a member of fstream, ifstream and ofstream objects in C ++:

```
void  open ( const char * ten_file ,  ios :: che_do );
```

Here, the first parameter specifies the name and location of the file to be opened and the second parameter of the **open ()** member function defines the mode in which the file should be opened.

Download mode :: Append Mode app. All output to that file is appended to the end of that file ios :: ate Open a file for outpur and move read / write control to the end of the ios :: file Open a file to read ios :: out Open a file to write ios :: trunc If this file already exists, its contents will be truncated before opening the file.
You can combine these two or more values ??together or together (use (|). For example, if you want to open a file in write mode and want to cut (truncate) it in case it has exists, you follow the following syntax:

```
ofstream outfile ;  outfile . open ( "file.dat" ,  ios :: out |  ios :: trunc
```

In the same way, you can open a file for the purpose of reading and writing as follows:

```
fstream QTM ;  QTM . open ( "file.dat" ,  ios :: out |  ios :: in );
```

Close a File in C ++

When a C ++ program finishes, it automatically closes all Streams, frees all allocated memory and closes all open files. But it is a good practice for a programmer to close all open files before the end of the program.

Here is the general syntax for **close ()** function in C ++, which is a member of fstream, ifstream and ofstream objects in C ++:

```
void  close ();
```

Record File in C ++

While programming C ++, you write information to a file from your program using the thread insertion operator (), just like when you use that operator to create output information to the screen. The only difference is that you use an object **ofstream** or **fstream** in C ++ instead of a **cout** object in C ++.

Read a File in C ++

You read information from a file in your C ++ program using the thread extraction operator (>>), just like you use that operator to enter input information from the keyboard. The difference is that you use an ifstream or fstream object instead of using the cin object in C ++.

Example of Reading and Writing Files in C ++

The following C ++ program opens a file in read and write mode. After writing the information entered by the user to a file qtm.dat, the program reads the information from that file and creates the output on the screen:

```
#include #include using namespace  std ; int  main  () { char  data [ 100 ]; //
```

Compiling and running the above C ++ program will produce the following results:

The example above uses additional functions from the cin object, like the getline () function to read the line from the outside and the ignore () function to ignore the extra characters to the left of the previous read command.

Location file pointer in C ++

Both istream and ostream objects provide member functions to redefine the file-position pointer. These member functions are **seekg** (seek get) for istream and **seekp** (short for seek put) for ostream in C ++.

The parameter for **seekg** and **seekp** is a **long int** . The second parameter can be specified to guide the search direction. The search direction can be ios :: beg (default) to locate the starting part of a **Stream** , ios :: cur to determine the location relative to the current location in a **Stream** or is ios :: end to determine the location relative to the end of a Stream in C ++.

The file location pointer is an integer value that determines the location in the file, calculating the number of bytes from the starting position of the file. Here are some examples to determine the location of file location pointers in C ++:

```
// xac dinh vi tri byte thu n cua doi tuong file  doi_tuong_file . seekg (  n
```

According to Tutorialspoint

Previous article: Interface in C ++ (Abstract class)

Next article: Exception handling (Exception Handling) in C ++