CodeWithHarry

HTML CSS JS C C++ JAVA PYTHON PHP

File I/O in C++: Read/Write in the Same Program & Closing Files | C++ Tutorials for Beginners #61

Overview Q&A Downloads Announcements

File I/O in C++: Read/Write in the Same Program & Closing Files | C++ Tutorials for Beginners #61

In this tutorial, we'll learn about creating a program that will read from a file and write to the file in the same program using a constructor.

Before jumping on to the main thing, we'll first give ourselves a quick revision of the things we had learned previously.

We had learned about the three most useful classes when we talk about File I/O, namely,

1. fstreambase
2. ifstream
3. ofstream.

All the above three classes can be used in a program by first including the header file, fstream.

Reading File Operation Output*:*

We learnt reading from a file using ifstream. Below snippet will help you recollect the same.

```
  string st;
// Opening files using constructor and reading it
  ifstream in("this.txt"); // Read operation
  in>>st;
```

Writing File Operation Output*:*

We learnt reading from a file using ofstream. Below snippet will help you recollect the same.

```
  string st = "Harry bhai";
  // Opening files using constructor and writing it
  ofstream out("this.txt"); // Write operation
  out<<st;
```

Let me make these codes functional in the same program for you to easily understand the workflow.
Suppose we have a file named sample60.txt in the same directory, we can easily call the file infinite number of times in the same program only by maintaining different connections for different purposes, using

```
<object_name>.close();
```

Now, let's move on to our systems. Open your editors as well. Don't forget to include the header file, <fstream>.
Follow these steps below to first write into the empty file:

1. Create a text file “sample60.txt” in the same directory as that of the program.
2. Create a string variable *name*.
3. Create an object *hout*(name it whatever you wish) using ofstream passing the text file, sample60.txt into it. This establishes a connection between your program and the text file.
4. Take input from the user using cin into the name string.(You can write manually as well)
5. Pass this name string to the object *hout.* The string name gets written in the text file.
6. Disconnect the file with the program since we are done writing to it using *hout.close().*

Since the file has been disconnected from the program, we can connect it again for any other purpose in the same program independently.

Follow these steps below to read from the file we just wrote into:

1. Create a string variable *content*.
2. Create an object *hin*(name it whatever you wish) using ifstream passing the text file, sample60.txt into it. This establishes a new connection between your program and the text file.
3. Fill in the string using the object *hin.* (Use getline, which we talked about in the last video, to take into input the whole line from the text file.)
4. Give output to the user, the string we filled in with the content in the text file.
5. Disconnect the file with the program since we are done reading from it using *hin.close().*

```
#include<iostream>
#include<fstream>

using namespace std;

int main(){
```

```
    // connecting our file with hout stream
     ofstream hout("sample60.txt");

     // creating a name string variable and filling it with string entered by the
    string name;
     cout<<"Enter your name: ";
    cin>>name;

   // writing a string to the file
   hout<<name + " is my name";

  // disconnecting our file
  hout.close();
  // connecting our file with hin stream
 ifstream hin("sample60.txt");

// creating a content string variable and filling it with string present there in
 string content;
  hin>>content;
 cout<<"The content of the file is: "<<content;

 // disconnecting our file
 hin.close();
 return 0;
}
```

Let's run the program we just created, The output will look like this:

```
Enter your name: Harry
The content of the file is: Harry
```

So when we input a string "Harry" into the text file, it gets written there in the file as below, and when we read it from the file, it gives output as below. Since we used hin and not getline, it could read just the first word.

```
The content of the file is: Harry
```

Thank you, friends for being with me throughout, hope you liked the tutorial. If you haven't checked out the whole playlist yet, move on to codewithharry.com or my YouTube channel to access it. I hope you enjoy them. In the next tutorial, we'll be covering the use of open() and eof() functions, see you there, till then keep coding.

Previous Next