**Error handling during file operation in C++**

It's quite common that errors may occur during file operations. There may have different reasons for arising errors while working with files. The following are the common problems that lead to errors during file operations.

* When trying to open a file for reading might not exist.
* When trying to read from a file beyond its total number of characters.
* When trying to perform a read operation from a file that has opened in write mode.
* When trying to perform a write operation on a file that has opened in reading mode.
* When trying to operate on a file that has not been open.

During the file operations in C++, the status of the current file stream stores in an integer flag defined in ios class. The following are the file stream flag states with meaning.

| **Flag Bit** | **Meaning** |
| --- | --- |
| badbit | 1 when a fatal I/O error has occurred, 0 otherwise. |
| failbit | 1 when a non-fatal I/O error has occurred, 0 otherwise |
| goodbit | 1 when no error has occurred, 0 otherwise |
| eofbit | 1 when end-of-file is encountered, 0 otherwise. |

We use the above flag bits to handle the errors during the file operations.

Error Handling During the File Operations in C++

The C++ programming language provides several built-in functions to handle errors during file operations.

The following are the built-in functions to handle file errors.

|  |  |
| --- | --- |
| Function | Return Value |
| int bad() | It returns a non-zero (true) value if an invalid operation is attempted or an unrecoverable error has occurred. Returns zero if it may be possible to recover from any other error reported and continue operations. |
| int fail( ) | It returns a non-zero (true) value when an input or output operation has failed. |
| int good() | It returns a non-zero (true) value when no error has occurred; otherwise returns zero (false). |
| int eof( ) | It returns a non-zero (true) value when end-of-file is encountered while reading; otherwise returns zero (false). |

int bad( )

The **bad( )** function returns a non-zero (true) value if an invalid operation is attempted or an unrecoverable error has occurred. Returns zero if it may be possible to recover from any other error reported and continue operations.

Let's look at the following code.

Example - Code to illustrate the bad( ) function

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

fstream file;

file.open("my\_file.txt", ios::out);

string data;

file >> data;

if(!file.bad()){

cout << "Operation not success!!!" << endl;

cout << "Status of the badbit: " << file.bad() << endl;

}

else {

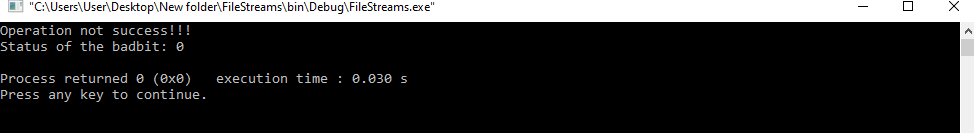
cout << "Data read from file - " << data << endl;

}

return 0;

}

Output



int fail( )

The **fail( )** function returns a non-zero value when an input or output operation has failed.

Let's look at the following code.

Example - Code to illustrate the fail( ) function

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

fstream file;

file.open("my\_file.txt", ios::out);

string data;

file >> data;

if(file.fail()){

cout << "Operation not success!!!" << endl;

cout << "Status of the failbit: " << file.fail() << endl;

}

else {

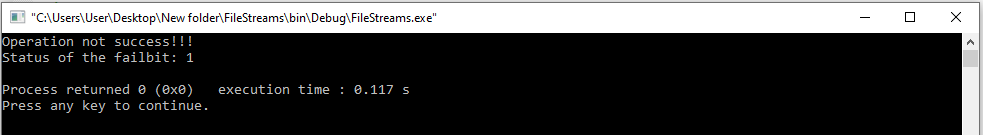
cout << "Data read from file - " << data << endl;

}

return 0;

}

Output



int eof( )

The **eof( )** function returns a non-zero (true) value when end-of-file is encountered while reading; otherwise returns zero (false).

Let's look at the following code.

Example - Code to illustrate the eof( ) function

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

fstream file;

file.open("my\_file.txt", ios::in);

string data;

while(!file.eof()){

file >> data;

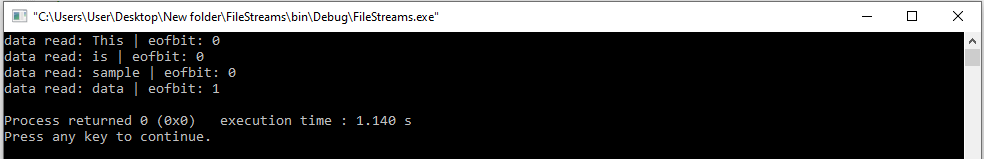
cout << "data read: " << data << " | eofbit: " << file.eof() << endl;

}

return 0;

}

Output



int good( )

The **good( )** function returns a non-zero (true) value when no error has occurred; otherwise returns zero (false).

Let's look at the following code.

Example - Code to illustrate the good( ) function

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

fstream file;

file.open("my\_file.txt", ios::in);

cout << "goodbit: " << file.good() << endl;

string data;

cout << endl << "Data read from file:" << endl;

while(!file.eof()){

file >> data;

cout << data << " ";

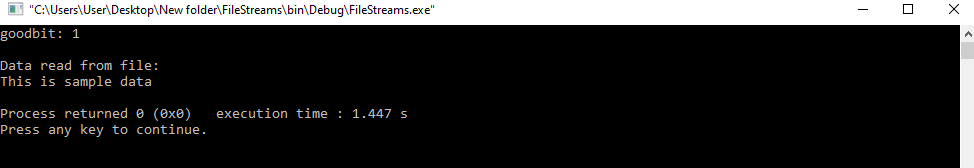
}

cout << endl;

return 0;

}

Output



int clear( )

The **clear( )** function used to reset the error state so that further operations can be attempted.

* The above functions can be summarized as eof() returns true if eofbit is set; bad() returns true if badbit is set. The fail() function returns true if failbit is set; the good() returns true there are no errors. Otherwise, they return false.
* All the built-in function returns either non-zero to indicate true or zero to indicate false.