Checking the file I/O status in Open C++

Overview

This code snippet shows how to get information about file I/O status in Open C++. The member function `rdstate()` returns the current internal error state flags of the stream (goodbit, eofbit, failbit, badbit). More than one error state could be set at the same time and it is possible to check for a specific state with the & operator and a specific bitmask. The member function `rdstate()` returns value goodbit if there are no errors. The other way to check the file I/O status is to use the member functions `good()`, `eof()`, `fail()`, and `bad()`. The member function `clear()` can be used to reset the state flags.

Note: In order to use this code, you need to install the Open C/C++ plug-in.

This snippet can be self-signed.

MMP file

The following libraries are required:

```
STATICLIBRARY libcrt0.lib

LIBRARY libstdcpp.lib
LIBRARY libc.lib
LIBRARY euser.lib
```

Source file

```
#include <fstream>  //fstream
#include <string>   //string, getline()
#include <iostream> //cout

using namespace std;

void check_stream_state1(const fstream &s)
{

    int state = s.rdstate();

    /* State flags:
    goodbit = 0x00,  //file has been opened without problems
    badbit  = 0x01,  //e.g. writing to a file that is not open / no space left
    eofbit  = 0x02,  //file opened for reading has reached the end
    failbit = 0x04,  //e.g. format error: alphabetical character is extracted
                     //from the stream but integer is expected
     */

    //goodbit is a constant with the value 0
    if (!state)
        {
        cout << "goodbit - no error occurred" << endl;
        }
    else
        {
        if (state & ios::badbit)
            cout << "badbit - a fatal I/O error has occurred" << endl;
        if (state & ios::eofbit)
            cout << "eofbit - end of file is encountered" << endl;
        if (state & ios::failbit)
            cout << "failbit - a nonfatal I/O error has occurred" << endl;
        }
}

void check_stream_state2(const fstream &s)
{
    //true if no error indicators are set
    if(s.good())
        {
        cout << "good() returned true" << endl;
        }
    else
        {
        //true if a stream is marked as unusable (badbit)
        if (s.bad())
            cout << "bad() returned true" << endl;
        //true if end of file is reached on the stream
        if (s.eof())
            cout << "eof() returned true" << endl;
        //true if some operation failed (failbit) or
        //the stream is marked as unusable (badbit)
        if(s.fail())
            cout << "fail() returned true" << endl;
        }
}

int main()
{
    fstream output("stream.txt", ios::out);

    //using member function good()
    if(output.good())
        {
```


(C) Copyright Nokia 2013. All rights reserved.

```
        //write text to the file
        output << "Text To File" << endl;
        }

    //check that goodbit is set
    check_stream_state1(output);
    check_stream_state2(output);

    //close the output stream
    output.close();

    //try to write text to a file that is not open
    output << "Some More Text" << endl;

    //check that badbit and failbit are set
    check_stream_state1(output);
    check_stream_state2(output);

    fstream input("stream.txt", ios::in);
    string text;

    //using member function eof()
    while(!input.eof())
        {
        getline (input, text);
        cout << text << endl;
        }

    //reset/set state flag to be goodbit
    input.clear();

    //try to write to input-only stream
    input << "Some More Text" << endl;

    //check that badbit and failbit are set
    check_stream_state1(input);
    check_stream_state2(input);

    //reset/set state flag to be goodbit
    input.clear();

    //move read pointer back to the beginning of the file
    input.seekg(0,ios::beg);

    int number = 0;

    //try to extract integer from the stream
    input >> number;

    //check that failbit is set
    check_stream_state1(input);
    check_stream_state2(input);

    //close the input stream
    output.close();

    //getchar();

    return 0;
}
```

Postconditions

The example application has created the file stream.txt containing one line of text. `check_stream_state()` functions have displayed state flag information about file I/O statuses as standard output.

See also

- Reading and writing text files in Open C++
- Reading and writing binary files in Open C++

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.