|  |  |
| --- | --- |
| **File open modes** | **Default Open Mode** |
| ofstream | The file is opened for output only. (Information may be written to the file, but not read from the file.) If the file does not exist, it is created. If the file already exists, its contents are deleted (the file is erased). |
| ifstream | The file is opened for input only. (Information may be read from the file, but not written to it.) The file’s contents will be read from its beginning. If the file does not  exist, the open function fails. |
| fstream | File open modes are predefined values that are members of the ios class. The ios::in mode is used to set the fstream object for input, and ios::out is used to set it for output. For example, we can open a file input.dat for input, and output.dat for output, using the fstream object and the appropriate file modes as follows: fstream inFile, outFile; inFile.open(“input.dat”, ios::in); outFile.open(“output.dat”, ios::out); If possible to combine several file open mode flags when a file is being opened. The combination of flags is achieved through the bitwise or operator |. Here is an example of combining the mode flags to open for both reading and writing: fstream dataFile; myFile.open(“myFile.dat”, ios::in | ios::out); |

//This program demonstrates the use of an fstream object and file mode flags (fstream\_mode1.cpp)

#include <iostream>  
#include <string>  
#include <fstream>

using namespace std;

int main()

{

fstream dataFile; // file object

string buffer; // Used to read line from file

// Create a new file named myfile.dat to write to.

dataFile.open("myfile.dat", ios::out);

// Write two lines to the file.

dataFile << "Now is the time for all good men" << endl

<< "to come to the aid of their country.";

// Close the file.

dataFile.close();

// Open the file for input.

dataFile.open("myfile.dat", ios::in);

// Read a line into a buffer and print the line.

getline(dataFile, buffer);

cout << buffer << endl;

// Read a second line and print it.

getline(dataFile, buffer);

cout << buffer << endl;

// Close the file.

dataFile.close();

system (“PAUSE”);

return 0;

}

|  |  |
| --- | --- |
| **File Mode Flag** | **Meaning** |
| ios::app | Append mode. If the file already exists, its contents are preserved and all output is written to the end of the file. By default, this flag causes the file to be created if it does not exist. |
| ios::ate | Initial output to the file will take place at the end of the file. |
| ios::binary | Binary mode. When a file is opened in binary mode, information is written to  or read from it in pure binary format. (The default mode is text.) |
| ios::in | Input mode. Information will be read from the file. If the file does not exist, it  will not be created and the open function will fail. |
| ios::out | Output mode. Information will be written to the file. By default, the file’s  contents will be deleted if it already exists. |
| ios::trunc | If the file already exists, its contents will be deleted (truncated). This is the  default mode used by ios: : out. |

// This program writes information to a file, closes the file,

// then reopens it and appends more information. (fstream\_mode2.cpp)

#include <fstream>

using namespace std;

int main()

{

fstream dataFile; // file object

// Open a file to write to, and write to it.

dataFile.open("demofile.txt", ios::out);

dataFile << "Jones\n";

dataFile << "Smith\n";

// Close the file.

dataFile.close();

// Open the same file in append mode, and write to it.

dataFile.open("demofile.txt", ios::out | ios::app);

dataFile << "Willis\n";

dataFile << "Davis\n";

// Close the file.

dataFile.close();

return 0;

}

// Program Area demonstrates simple stream testing

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

int side1; // one side of a rectangle

int side2; // the other side of a rectangle

ifstream inData; // file stream

int area; // area of rectangle

inData.open("myData.dat");

if (!inData)

{

cout << "Input file not found." << endl;

return 1;

}

inData >> side1 >> side2;

if (!inData)

{

cout << "Data format incorrect.";

return 2;

}

area = side1 \* side2;

cout << "Area is " << area << endl;

return 0;

}

CONTENTS OF DATA FILE myData.dat

5 5