

C++ has support both for input and output with files through the following classes:

- **ofstream**: File class for writing operations (derived from ostream)
- **ifstream**: File class for reading operations (derived from istream)
- **fstream**: File class for both reading and writing operations (derived from iostream)

● Open a file

The first operation generally done on an object of one of these classes is to associate it to a real file, that is to say, to open a file. The open file is represented within the program by a stream object (an instantiation of one of these classes) and any input or output performed on this stream object will be applied to the physical file.

In order to open a file with a stream object we use its member function **open()**:

```
void open (const char * filename, openmode mode);
```

where *filename* is a string of characters representing the name of the file to be opened and *mode* is a combination of the following flags:

ios::in	Open file for reading
ios::out	Open file for writing
ios::ate	Initial position: end of file
ios::app	Every output is appended at the end of file
ios::trunc	If the file already existed it is erased
ios::binary	Binary mode

These flags can be combined using bitwise operator OR: |. For example, if we want to open the file "example.bin" in binary mode to add data we could do it by the following call to function-member **open**:

```
ofstream file;
```

```
file.open ("example.bin", ios::out | ios::app | ios::binary);
```

All of the member functions **open** of classes **ofstream**, **ifstream** and **fstream** include a default mode when opening files that varies from one to the other:

Class	Default <i>mode</i> to parameter
ofstream	ios::out ios::trunc
ifstream	ios::in
fstream	ios::in ios::out

The default value is only applied if the function is called without specifying a *mode* parameter. If the function is called with any value in that parameter the default mode is stepped on, not combined.

Since the first task that is performed on an object of classes **ofstream**, **ifstream** and **fstream** is frequently to open a file, these three classes include a constructor that directly calls the **open** member function and has the same parameters as this. This way, we could also have declared the previous object and conducted the same opening operation just by writing:

```
ofstream file ("example.bin", ios::out | ios::app | ios::binary);
```

Both forms to open a file are valid.

You can check if a file has been correctly opened by calling the member function **is_open()**:

```
bool is_open();
```

that returns a **bool** type value indicating **true** in case that indeed the object has been correctly associated with an open file or **false** otherwise.

● Close a file

When reading, writing or consulting operations on a file are complete we must close it so that it becomes available again. In order to do that we shall call the member function **close()**, that is in charge of flushing the buffers and closing the file. Its form is quite simple:

```
void close ();
```

Once this member function is called, the stream object can be used to open another file, and the file is available again to be opened by other processes.

In case that an object is destructed while still associated with an open file, the destructor automatically calls the member function **close**.

● Test mode files

Classes **ofstream**, **ifstream** and **fstream** are derived from **ostream**, **istream** and **iostream** respectively. That's why *fstream* objects can use the members of these parent classes to access data.

Generally, when using text files we shall use the same members of these classes that we used in communication with the console (**cin** and **cout**). As in the following example, where we use the overloaded insertion operator **<<**:

Examples:

```
// writing on a text file
#include <fstream.h>
int main () {
    ofstream examplefile ("example.txt");
    if (examplefile.is_open())
    {
        examplefile << "AAA\n";
        examplefile << "BBB\n";
        examplefile.close();
    }
}
```

```
    }  
    return 0;  
}  
  
// reading a text file  
#include <iostream.h>  
#include <fstream.h>  
#include <stdlib.h>  
  
int main () {  
    char buffer[256];  
    ifstream examplefile ("example.txt");  
    if (! examplefile.is_open())  
        { cout << "Error opening file"; exit (1); }  
  
    while (! examplefile.eof() )  
        {  
            examplefile.getline (buffer,100);  
            cout << buffer << endl;  
        }  
    return 0;  
}
```