**File Handling in** c++

**Define:**

* A file is a collection of related records.
* A records contains data about an entity.
* A file can be used to save any type of data.
* Files are stored on secondary storage.
* The data is stored permanently.

**Types of files:**

* Text files
* Binary files

**Stream:**

Stream is a series of bytes associated with a file.it consists of data that is transferred from to another one location.

**Here we will learn the following things one by one.**

1. What are Headers in C++
2. How to open and close a file
3. How to write into a file in C++
4. Read data from a file
5. Some important operations on file handling in C++

Headers

Along with the “instream” header, we also import a header called “stream”.

It is used for enabling the file handling function in C++  
IOSTREAM = Input Output Stream. Similarly,  
FSTREAM = File Stream

#include <instream>

#include <stream>

**This package helps us use 3 new functions**

1. of stream (Output File Stream): Used for basic Output functions and writing into a file
2. if stream (Input File Stream): Used for basic Input functions and reading from a file
3. stream (File Stream): Used for both writing and reading from a file

Along with these 3 functions, we also get 4 different operations we can use on a file

1. open (): Used to Open a File
2. read (): Used to Read the File
3. write (): Used to Write data into the File
4. close (): Used to Close the File

### Opening/Closing a File in C++

#include <instream>

#include <stream>

using namespace stud;

in main ()

{

stream ob.;

obj. open ("C:\File1212.txt”, iOS: out);

if (! ob.)

{

court<<"File exists";

}

else

{

court<<"file created";

obj. close ();

}

return 0;

}

This code is used to open a file.  
But what if the file at that location does not exist?  
The compiler automatically creates a new file over there with the name File1212 i.e. the name given to your file

**Now let us address some important lines in the code**

1. “stream ob.”;  :  
   Here we are creating an object “ob.” that is used to refer to the function “stream” we are using. It is basically a file pointer
2. obj.open(“C:\File1212.txt”,ios::out);  :  
   Here we are using the function “open” to tell the compiler to open the file at the location we are entering. The iOS is used to tell the compiler what operation we would be performing on the file.  
   It has 4 modes:  
   1) iOS::out  :  To use for file writing  
   2) iOS::in  :  To use for file inputting  
   3) iOS::app  :  To use for appending a file  
   4) iOS::trunk  :  To use for truncating a file  
   5) iOS::beg  :  To tell the beginning point of the file  
   6)iOS::cur  : To tell the current position of the pointer  
   7)iOS::end : To tell the endpoint of the file
3. if (! ob.)  :  
   Here we are checking if the file already exists
4. obj.close();  :  
   If the file is opened, it’ll have to be closed too

**Hence a file is created**

### Writing into a file in C++

#include <instream>

#include <stream>

using namespace stud;

in main ()

{

stream ob.;

obj. open ("C: File.txt”, IOS: out);

if (! ob.)

{

court<<"could not create file";

}

else

{

ob.<<"Hello World";

obj.close();

}

return 0;

}

If you notice carefully, there is only one-line difference between the previous code and this code i.e.

ob.<< “Hello World”

So basically we are writing hello world into the file that ob. points to.  
Do try this code with your preferred file location and file name. Then open that file and notice the text written out there

### Reading from a File in C++

#include <instream>

#include <stream>

using namespace std;

int main()

{

fstream obj;

obj.open("C:File.txt",ios::in);

if(!obj)

{

cout<<"could not create file";

}

else

{

char chi;

while (! obj.eof ())

{

obj>>chi;

court<<chi<<" ";

}

obj. close ();

}

return 0;

}

Again, this code is mostly similar to the previous codes except for a few lines  
I have written “Testing” into the file and I am reading it and printing it with a space between each character.  
Notice, we have used ios::in for reading

1. while(!obj.eof())  :  
   Here we are creating a loop that iterates the number of times a character is there.  
   elf() basically means end of file. It is when the pointer reaches the point after the last character in the file
2. obj>>chi;  :  
   Here we are extracting a character and storing it in chi
3. court<<chi<<” “;  :  
   Here we are printing the extracted character and printing it.

Hence the output would be:

### Additional Operations on file handing

1. tell():  
   Tells the current put pointer’s location
2. tell():  
   Tells the current get pointer’s location
3. seek():  
   Moves the put pointer to the desired location
4. seek():  
   Moves the get pointer to the desired location
5. put():  
   Write a single line of character
6. get():  
   Read a single line of character