**How to Copy a File to another File in Java**

In this tutorial we will see how to copy the content of one file to another file in java. In order to copy the file, first we can [**read the file**](http://beginnersbook.com/2014/01/how-to-read-file-in-java-bufferedinputstream/) using[**FileInputStream**](http://docs.oracle.com/javase/6/docs/api/java/io/FileInputStream.html) and then we can [**write the read content to the output file**](http://beginnersbook.com/2014/01/how-to-write-to-a-file-in-java-using-fileoutputstream/)using **[FileOutputStream](http://docs.oracle.com/javase/6/docs/api/java/io/FileOutputStream.html" \t "_blank)**.

**Example**

The below code would copy the content of “MyInputFile.txt” to the “MyOutputFile.txt” file. If “MyOutputFile.txt” doesn’t exist then the program would create the file first and then it would copy the content.

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class CopyExample

{

public static void main(String[] args)

{

FileInputStream instream = null;

FileOutputStream outstream = null;

try{

File infile =new File("C:\\MyInputFile.txt");

File outfile =new File("C:\\MyOutputFile.txt");

instream = new FileInputStream(infile);

outstream = new FileOutputStream(outfile);

byte[] buffer = new byte[1024];

int length;

/\*copying the contents from input stream to

\* output stream using read and write methods

\*/

while ((length = instream.read(buffer)) > 0){

outstream.write(buffer, 0, length);

}

//Closing the input/output file streams

instream.close();

outstream.close();

System.out.println("File copied successfully!!");

}catch(IOException ioe){

ioe.printStackTrace();

}

}

}

Output:

File copied successfully!!

The methods used in the above program are:

**read() method**

public int read(byte[] b) throws IOException

Reads up to b.length bytes of data from this input stream into an array of bytes. This method blocks until some input is available. It returns total number of bytes read into the buffer, or -1 if there is no more data because the end of the file has been reached. In order to make this method work in our program, we have created a byte array “buffer” and reading the content of input file to the same. Since this method throws IOException, we have put the “read file” code inside try-catch block in order to handle the exception.

**write() method**

public void write(byte[] b,

int off,

int length)

throws IOException

Writes length bytes from the specified byte array starting at offset off to this file output stream.

**Tweaks:**  
If your input and outfile files are not in the same drive then you can specify the drive while creating the file object. For example if your input file is in C drive and output file is in D drive then you can create the file object like this:

File infile =new File("C:\\MyInputFile.txt");

File outfile =new File("D:\\MyOutputFile.txt");