**How to write to file in Java using BufferedWriter**

[**JAVA I/O**](http://beginnersbook.com/category/java-io/)

Earlier we discussed how to [**write to a file using FileOutputStream**](http://beginnersbook.com/2014/01/how-to-write-to-a-file-in-java-using-fileoutputstream/). In this tutorial we will see how to write to a file using BufferedWriter. We will be using write() method of BufferedWriter to write the text into a file. The advantage of using BufferedWriter is that it writes text to a character-output stream, buffering characters so as to provide for the efficient writing (better performance) of single characters, arrays, and strings.

**Complete example: Write to file using BufferedWriter**

In this example we have a String mycontent and a file myfile.txt in C drive. We are writing the String to the File with the help of FileWriter and BufferedWriter.

package beginnersbook.com;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

public class WriteFileDemo {

public static void main(String[] args) {

BufferedWriter bw = null;

try {

String mycontent = "This String would be written" +

" to the specified File";

//Specify the file name and path here

File file = new File("C:/myfile.txt");

/\* This logic will make sure that the file

\* gets created if it is not present at the

\* specified location\*/

if (!file.exists()) {

file.createNewFile();

}

FileWriter fw = new FileWriter(file);

bw = new BufferedWriter(fw);

bw.write(mycontent);

System.out.println("File written Successfully");

} catch (IOException ioe) {

ioe.printStackTrace();

}

finally

{

try{

if(bw!=null)

bw.close();

}catch(Exception ex){

System.out.println("Error in closing the BufferedWriter"+ex);

}

}

}

}

Output:

File written Successfully