One of the most frequently used task in programming is writing to and reading from a file. To do this in Java there are more possibilities. At this time only the most frequently used text file handling solutions will be presented.

Filename handling

To write anything to a file first of all we need a file name we want to use. The file name is a simple string like like this:

- *String fileName = "test.txt";*

If you want to write in a file which is located elsewhere you need to define the complete file name and path in your fileName variable:

- *String fileName = "c:\\filedemo\\test.txt";*

However if you define a path in your file name then you have to take care the path separator. On windows system the '\' is used and you need to backslash it so you need to write '\\', in Unix,Linux systems
the separator is a simple slash '/'.

To make your code OS independent you can get the separator character as follows:

- *String separator = File.separator;*

Open a file

To open a file for writing use the FileWriter class and create an instance from it. The file name is passed in the constructor like this:

- *FileWriter writer = new FileWriter(fileName);*

This code opens the file in overwrite mode. If you want to append to the file then you need to use an other constructor like this:

- *FileWriter writer = new FileWriter(fileName,true);*

Besides this the constructor can throw an IOException so we put all of the code inside a try-catch block.

Write to file

At this point we have a writer object and we can send real content to the file. You can do this using the **write()** method, which has more variant but the most commonly used requires a

string as input parameter.

Calling the **write()** method doesn't mean that it immediately writes the data into the file. The output is maybe cached so if you want to send your data immediately to the file you need to call the **flush()** method.

As last step you should close the file with the **close()** method and you are done.

The basic write method looks like this:

```
1. public void writeFile() {
2.     String fileName = "c:\\test.txt";
3.
4.     try {
5.         FileWriter writer = new FileWriter(fileName,true);
6.         writer.write("Test text.");
7.         writer.close();
8.     } catch (IOException e) {
9.         e.printStackTrace();
10.          }
11.     }
```

Java package

A Java package is a mechanism for organizing Java classes into namespaces similar to the modules of Modula. Java packages can be stored in compressed files called JAR files, allowing classes to download faster as a group rather than one at a time. Programmers also typically use packages to organize classes belonging to the same category or providing similar functionality.

- A package provides a unique namespace for the types it contains.
- Classes in the same package can access each other's package-access members

In a Java source file, the package that this file's class or classes belong to is specified with the `package` keyword. This keyword is usually the first keyword in the source file.

```
package java.awt.event;
```

To use a package's classes inside a Java source file, it is convenient to import the classes from the package with an `import` declaration. The following declaration

```
import java.awt.event.*;
```