| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/FileHandler.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/logging/ErrorManager.html)   [**NEXT CLASS**](http://docs.google.com/java/util/logging/Filter.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/logging/FileHandler.html)    [**NO FRAMES**](http://docs.google.com/FileHandler.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#35nkun2) |

## **java.util.logging**

Class FileHandler

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.util.logging.Handler](http://docs.google.com/java/util/logging/Handler.html)  
 [java.util.logging.StreamHandler](http://docs.google.com/java/util/logging/StreamHandler.html)  
 **java.util.logging.FileHandler**

public class **FileHandler**extends [StreamHandler](http://docs.google.com/java/util/logging/StreamHandler.html)

Simple file logging Handler.

The FileHandler can either write to a specified file, or it can write to a rotating set of files.

For a rotating set of files, as each file reaches a given size limit, it is closed, rotated out, and a new file opened. Successively older files are named by adding "0", "1", "2", etc into the base filename.

By default buffering is enabled in the IO libraries but each log record is flushed out when it is complete.

By default the XMLFormatter class is used for formatting.

**Configuration:** By default each FileHandler is initialized using the following LogManager configuration properties. If properties are not defined (or have invalid values) then the specified default values are used.

* java.util.logging.FileHandler.level specifies the default level for the Handler (defaults to Level.ALL).
* java.util.logging.FileHandler.filter specifies the name of a Filter class to use (defaults to no Filter).
* java.util.logging.FileHandler.formatter specifies the name of a Formatter class to use (defaults to java.util.logging.XMLFormatter)
* java.util.logging.FileHandler.encoding the name of the character set encoding to use (defaults to the default platform encoding).
* java.util.logging.FileHandler.limit specifies an approximate maximum amount to write (in bytes) to any one file. If this is zero, then there is no limit. (Defaults to no limit).
* java.util.logging.FileHandler.count specifies how many output files to cycle through (defaults to 1).
* java.util.logging.FileHandler.pattern specifies a pattern for generating the output file name. See below for details. (Defaults to "%h/java%u.log").
* java.util.logging.FileHandler.append specifies whether the FileHandler should append onto any existing files (defaults to false).

A pattern consists of a string that includes the following special components that will be replaced at runtime:

* "/" the local pathname separator
* "%t" the system temporary directory
* "%h" the value of the "user.home" system property
* "%g" the generation number to distinguish rotated logs
* "%u" a unique number to resolve conflicts
* "%%" translates to a single percent sign "%"

If no "%g" field has been specified and the file count is greater than one, then the generation number will be added to the end of the generated filename, after a dot.

Thus for example a pattern of "%t/java%g.log" with a count of 2 would typically cause log files to be written on Solaris to /var/tmp/java0.log and /var/tmp/java1.log whereas on Windows 95 they would be typically written to C:\TEMP\java0.log and C:\TEMP\java1.log

Generation numbers follow the sequence 0, 1, 2, etc.

Normally the "%u" unique field is set to 0. However, if the FileHandler tries to open the filename and finds the file is currently in use by another process it will increment the unique number field and try again. This will be repeated until FileHandler finds a file name that is not currently in use. If there is a conflict and no "%u" field has been specified, it will be added at the end of the filename after a dot. (This will be after any automatically added generation number.)

Thus if three processes were all trying to log to fred%u.%g.txt then they might end up using fred0.0.txt, fred1.0.txt, fred2.0.txt as the first file in their rotating sequences.

Note that the use of unique ids to avoid conflicts is only guaranteed to work reliably when using a local disk file system.

**Since:** 1.4

| **Constructor Summary** | |
| --- | --- |
| [**FileHandler**](http://docs.google.com/java/util/logging/FileHandler.html#FileHandler())()            Construct a default FileHandler. |
| [**FileHandler**](http://docs.google.com/java/util/logging/FileHandler.html#FileHandler(java.lang.String))([String](http://docs.google.com/java/lang/String.html) pattern)            Initialize a FileHandler to write to the given filename. |
| [**FileHandler**](http://docs.google.com/java/util/logging/FileHandler.html#FileHandler(java.lang.String,%20boolean))([String](http://docs.google.com/java/lang/String.html) pattern, boolean append)            Initialize a FileHandler to write to the given filename, with optional append. |
| [**FileHandler**](http://docs.google.com/java/util/logging/FileHandler.html#FileHandler(java.lang.String,%20int,%20int))([String](http://docs.google.com/java/lang/String.html) pattern, int limit, int count)            Initialize a FileHandler to write to a set of files. |
| [**FileHandler**](http://docs.google.com/java/util/logging/FileHandler.html#FileHandler(java.lang.String,%20int,%20int,%20boolean))([String](http://docs.google.com/java/lang/String.html) pattern, int limit, int count, boolean append)            Initialize a FileHandler to write to a set of files with optional append. |

| **Method Summary** | |
| --- | --- |
| void | [**close**](http://docs.google.com/java/util/logging/FileHandler.html#close())()            Close all the files. |
| void | [**publish**](http://docs.google.com/java/util/logging/FileHandler.html#publish(java.util.logging.LogRecord))([LogRecord](http://docs.google.com/java/util/logging/LogRecord.html) record)            Format and publish a LogRecord. |

| **Methods inherited from class java.util.logging.**[**StreamHandler**](http://docs.google.com/java/util/logging/StreamHandler.html) |
| --- |
| [flush](http://docs.google.com/java/util/logging/StreamHandler.html#flush()), [isLoggable](http://docs.google.com/java/util/logging/StreamHandler.html#isLoggable(java.util.logging.LogRecord)), [setEncoding](http://docs.google.com/java/util/logging/StreamHandler.html#setEncoding(java.lang.String)), [setOutputStream](http://docs.google.com/java/util/logging/StreamHandler.html#setOutputStream(java.io.OutputStream)) |

| **Methods inherited from class java.util.logging.**[**Handler**](http://docs.google.com/java/util/logging/Handler.html) |
| --- |
| [getEncoding](http://docs.google.com/java/util/logging/Handler.html#getEncoding()), [getErrorManager](http://docs.google.com/java/util/logging/Handler.html#getErrorManager()), [getFilter](http://docs.google.com/java/util/logging/Handler.html#getFilter()), [getFormatter](http://docs.google.com/java/util/logging/Handler.html#getFormatter()), [getLevel](http://docs.google.com/java/util/logging/Handler.html#getLevel()), [reportError](http://docs.google.com/java/util/logging/Handler.html#reportError(java.lang.String,%20java.lang.Exception,%20int)), [setErrorManager](http://docs.google.com/java/util/logging/Handler.html#setErrorManager(java.util.logging.ErrorManager)), [setFilter](http://docs.google.com/java/util/logging/Handler.html#setFilter(java.util.logging.Filter)), [setFormatter](http://docs.google.com/java/util/logging/Handler.html#setFormatter(java.util.logging.Formatter)), [setLevel](http://docs.google.com/java/util/logging/Handler.html#setLevel(java.util.logging.Level)) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Constructor Detail** |
| --- |

### FileHandler

public **FileHandler**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Construct a default FileHandler. This will be configured entirely from LogManager properties (or their default values).

**Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there are IO problems opening the files. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control")). [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if pattern property is an empty String.

### FileHandler

public **FileHandler**([String](http://docs.google.com/java/lang/String.html) pattern)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Initialize a FileHandler to write to the given filename.

The FileHandler is configured based on LogManager properties (or their default values) except that the given pattern argument is used as the filename pattern, the file limit is set to no limit, and the file count is set to one.

There is no limit on the amount of data that may be written, so use this with care.

**Parameters:**pattern - the name of the output file **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there are IO problems opening the files. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control"). [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if pattern is an empty string

### FileHandler

public **FileHandler**([String](http://docs.google.com/java/lang/String.html) pattern,  
 boolean append)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Initialize a FileHandler to write to the given filename, with optional append.

The FileHandler is configured based on LogManager properties (or their default values) except that the given pattern argument is used as the filename pattern, the file limit is set to no limit, the file count is set to one, and the append mode is set to the given append argument.

There is no limit on the amount of data that may be written, so use this with care.

**Parameters:**pattern - the name of the output fileappend - specifies append mode **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there are IO problems opening the files. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control"). [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if pattern is an empty string

### FileHandler

public **FileHandler**([String](http://docs.google.com/java/lang/String.html) pattern,  
 int limit,  
 int count)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Initialize a FileHandler to write to a set of files. When (approximately) the given limit has been written to one file, another file will be opened. The output will cycle through a set of count files.

The FileHandler is configured based on LogManager properties (or their default values) except that the given pattern argument is used as the filename pattern, the file limit is set to the limit argument, and the file count is set to the given count argument.

The count must be at least 1.

**Parameters:**pattern - the pattern for naming the output filelimit - the maximum number of bytes to write to any one filecount - the number of files to use **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there are IO problems opening the files. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control"). [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if limit < 0, or count < 1. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if pattern is an empty string

### FileHandler

public **FileHandler**([String](http://docs.google.com/java/lang/String.html) pattern,  
 int limit,  
 int count,  
 boolean append)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Initialize a FileHandler to write to a set of files with optional append. When (approximately) the given limit has been written to one file, another file will be opened. The output will cycle through a set of count files.

The FileHandler is configured based on LogManager properties (or their default values) except that the given pattern argument is used as the filename pattern, the file limit is set to the limit argument, and the file count is set to the given count argument, and the append mode is set to the given append argument.

The count must be at least 1.

**Parameters:**pattern - the pattern for naming the output filelimit - the maximum number of bytes to write to any one filecount - the number of files to useappend - specifies append mode **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there are IO problems opening the files. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control"). [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if limit < 0, or count < 1. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if pattern is an empty string

| **Method Detail** |
| --- |

### publish

public void **publish**([LogRecord](http://docs.google.com/java/util/logging/LogRecord.html) record)

Format and publish a LogRecord.

**Overrides:**[publish](http://docs.google.com/java/util/logging/StreamHandler.html#publish(java.util.logging.LogRecord)) in class [StreamHandler](http://docs.google.com/java/util/logging/StreamHandler.html) **Parameters:**record - description of the log event. A null record is silently ignored and is not published

### close

public void **close**()  
 throws [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Close all the files.

**Overrides:**[close](http://docs.google.com/java/util/logging/StreamHandler.html#close()) in class [StreamHandler](http://docs.google.com/java/util/logging/StreamHandler.html) **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and if the caller does not have LoggingPermission("control").

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/FileHandler.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/logging/ErrorManager.html)   [**NEXT CLASS**](http://docs.google.com/java/util/logging/Filter.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/logging/FileHandler.html)    [**NO FRAMES**](http://docs.google.com/FileHandler.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#35nkun2) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).