| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StandardJavaFileManager.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/javax/tools/SimpleJavaFileObject.html)   [**NEXT CLASS**](http://docs.google.com/javax/tools/StandardLocation.html) | [**FRAMES**](http://docs.google.com/index.html?javax/tools/StandardJavaFileManager.html)    [**NO FRAMES**](http://docs.google.com/StandardJavaFileManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#4d34og8) |

## **javax.tools**

Interface StandardJavaFileManager

**All Superinterfaces:** [Closeable](http://docs.google.com/java/io/Closeable.html), [Flushable](http://docs.google.com/java/io/Flushable.html), [JavaFileManager](http://docs.google.com/javax/tools/JavaFileManager.html), [OptionChecker](http://docs.google.com/javax/tools/OptionChecker.html)

public interface **StandardJavaFileManager**extends [JavaFileManager](http://docs.google.com/javax/tools/JavaFileManager.html)

File manager based on [java.io.File](http://docs.google.com/java/io/File.html). A common way to obtain an instance of this class is using [getStandardFileManager](http://docs.google.com/javax/tools/JavaCompiler.html#getStandardFileManager(javax.tools.DiagnosticListener,%20java.util.Locale,%20java.nio.charset.Charset)), for example:

JavaCompiler compiler = ToolProvider.getSystemJavaCompiler();  
 DiagnosticCollector<JavaFileObject> diagnostics =  
 new DiagnosticCollector<JavaFileObject>();  
 StandardJavaFileManager fm = compiler.getStandardFileManager(diagnostics, null, null);

This file manager creates file objects representing regular [files](http://docs.google.com/java/io/File.html), [zip file entries](http://docs.google.com/java/util/zip/ZipEntry.html), or entries in similar file system based containers. Any file object returned from a file manager implementing this interface must observe the following behavior:

* File names need not be canonical.
* For file objects representing regular files
  + the method [FileObject.delete()](http://docs.google.com/javax/tools/FileObject.html#delete()) is equivalent to [File.delete()](http://docs.google.com/java/io/File.html#delete()),
  + the method [FileObject.getLastModified()](http://docs.google.com/javax/tools/FileObject.html#getLastModified()) is equivalent to [File.lastModified()](http://docs.google.com/java/io/File.html#lastModified()),
  + the methods [FileObject.getCharContent(boolean)](http://docs.google.com/javax/tools/FileObject.html#getCharContent(boolean)), [FileObject.openInputStream()](http://docs.google.com/javax/tools/FileObject.html#openInputStream()), and [FileObject.openReader(boolean)](http://docs.google.com/javax/tools/FileObject.html#openReader(boolean)) must succeed if the following would succeed (ignoring encoding issues):new [FileInputStream](http://docs.google.com/java/io/FileInputStream.html#FileInputStream(java.io.File))(new [File](http://docs.google.com/java/io/File.html#File(java.net.URI))([fileObject](http://docs.google.com/javax/tools/FileObject.html).[toUri](http://docs.google.com/javax/tools/FileObject.html#toUri())()))
  + and the methods [FileObject.openOutputStream()](http://docs.google.com/javax/tools/FileObject.html#openOutputStream()), and [FileObject.openWriter()](http://docs.google.com/javax/tools/FileObject.html#openWriter()) must succeed if the following would succeed (ignoring encoding issues):new [FileOutputStream](http://docs.google.com/java/io/FileOutputStream.html#FileOutputStream(java.io.File))(new [File](http://docs.google.com/java/io/File.html#File(java.net.URI))([fileObject](http://docs.google.com/javax/tools/FileObject.html).[toUri](http://docs.google.com/javax/tools/FileObject.html#toUri())()))
* The [URI](http://docs.google.com/java/net/URI.html) returned from [FileObject.toUri()](http://docs.google.com/javax/tools/FileObject.html#toUri())
  + must be [absolute](http://docs.google.com/java/net/URI.html#isAbsolute()) (have a schema), and
  + must have a [normalized](http://docs.google.com/java/net/URI.html#normalize()) [path component](http://docs.google.com/java/net/URI.html#getPath()) which can be resolved without any process-specific context such as the current directory (file names must be absolute).

According to these rules, the following URIs, for example, are allowed:

* file:///C:/Documents%20and%20Settings/UncleBob/BobsApp/Test.java
* jar:///C:/Documents%20and%20Settings/UncleBob/lib/vendorA.jar!com/vendora/LibraryClass.class

Whereas these are not (reason in parentheses):

* file:BobsApp/Test.java (the file name is relative and depend on the current directory)
* jar:lib/vendorA.jar!com/vendora/LibraryClass.class (the first half of the path depends on the current directory, whereas the component after ! is legal)
* Test.java (this URI depends on the current directory and does not have a schema)
* jar:///C:/Documents%20and%20Settings/UncleBob/BobsApp/../lib/vendorA.jar!com/vendora/LibraryClass.class (the path is not normalized)

**Since:** 1.6

| **Nested Class Summary** | |
| --- | --- |

| **Nested classes/interfaces inherited from interface javax.tools.**[**JavaFileManager**](http://docs.google.com/javax/tools/JavaFileManager.html) |
| --- |
| [JavaFileManager.Location](http://docs.google.com/javax/tools/JavaFileManager.Location.html) |

| **Method Summary** | |
| --- | --- |
| [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> | [**getJavaFileObjects**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getJavaFileObjects(java.io.File...))([File](http://docs.google.com/java/io/File.html)... files)            Gets file objects representing the given files. |
| [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> | [**getJavaFileObjects**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getJavaFileObjects(java.lang.String...))([String](http://docs.google.com/java/lang/String.html)... names)            Gets file objects representing the given file names. |
| [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> | [**getJavaFileObjectsFromFiles**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getJavaFileObjectsFromFiles(java.lang.Iterable))([Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> files)            Gets file objects representing the given files. |
| [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> | [**getJavaFileObjectsFromStrings**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getJavaFileObjectsFromStrings(java.lang.Iterable))([Iterable](http://docs.google.com/java/lang/Iterable.html)<[String](http://docs.google.com/java/lang/String.html)> names)            Gets file objects representing the given file names. |
| [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> | [**getLocation**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getLocation(javax.tools.JavaFileManager.Location))([JavaFileManager.Location](http://docs.google.com/javax/tools/JavaFileManager.Location.html) location)            Gets the path associated with the given location. |
| boolean | [**isSameFile**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#isSameFile(javax.tools.FileObject,%20javax.tools.FileObject))([FileObject](http://docs.google.com/javax/tools/FileObject.html) a, [FileObject](http://docs.google.com/javax/tools/FileObject.html) b)            Compares two file objects and return true if they represent the same canonical file, zip file entry, or entry in any file system based container. |
| void | [**setLocation**](http://docs.google.com/javax/tools/StandardJavaFileManager.html#setLocation(javax.tools.JavaFileManager.Location,%20java.lang.Iterable))([JavaFileManager.Location](http://docs.google.com/javax/tools/JavaFileManager.Location.html) location, [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> path)            Associates the given path with the given location. |

| **Methods inherited from interface javax.tools.**[**JavaFileManager**](http://docs.google.com/javax/tools/JavaFileManager.html) |
| --- |
| [close](http://docs.google.com/javax/tools/JavaFileManager.html#close()), [flush](http://docs.google.com/javax/tools/JavaFileManager.html#flush()), [getClassLoader](http://docs.google.com/javax/tools/JavaFileManager.html#getClassLoader(javax.tools.JavaFileManager.Location)), [getFileForInput](http://docs.google.com/javax/tools/JavaFileManager.html#getFileForInput(javax.tools.JavaFileManager.Location,%20java.lang.String,%20java.lang.String)), [getFileForOutput](http://docs.google.com/javax/tools/JavaFileManager.html#getFileForOutput(javax.tools.JavaFileManager.Location,%20java.lang.String,%20java.lang.String,%20javax.tools.FileObject)), [getJavaFileForInput](http://docs.google.com/javax/tools/JavaFileManager.html#getJavaFileForInput(javax.tools.JavaFileManager.Location,%20java.lang.String,%20javax.tools.JavaFileObject.Kind)), [getJavaFileForOutput](http://docs.google.com/javax/tools/JavaFileManager.html#getJavaFileForOutput(javax.tools.JavaFileManager.Location,%20java.lang.String,%20javax.tools.JavaFileObject.Kind,%20javax.tools.FileObject)), [handleOption](http://docs.google.com/javax/tools/JavaFileManager.html#handleOption(java.lang.String,%20java.util.Iterator)), [hasLocation](http://docs.google.com/javax/tools/JavaFileManager.html#hasLocation(javax.tools.JavaFileManager.Location)), [inferBinaryName](http://docs.google.com/javax/tools/JavaFileManager.html#inferBinaryName(javax.tools.JavaFileManager.Location,%20javax.tools.JavaFileObject)), [list](http://docs.google.com/javax/tools/JavaFileManager.html#list(javax.tools.JavaFileManager.Location,%20java.lang.String,%20java.util.Set,%20boolean)) |

| **Methods inherited from interface javax.tools.**[**OptionChecker**](http://docs.google.com/javax/tools/OptionChecker.html) |
| --- |
| [isSupportedOption](http://docs.google.com/javax/tools/OptionChecker.html#isSupportedOption(java.lang.String)) |

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

### isSameFile

boolean **isSameFile**([FileObject](http://docs.google.com/javax/tools/FileObject.html) a,  
 [FileObject](http://docs.google.com/javax/tools/FileObject.html) b)

Compares two file objects and return true if they represent the same canonical file, zip file entry, or entry in any file system based container.

**Specified by:**[isSameFile](http://docs.google.com/javax/tools/JavaFileManager.html#isSameFile(javax.tools.FileObject,%20javax.tools.FileObject)) in interface [JavaFileManager](http://docs.google.com/javax/tools/JavaFileManager.html) **Parameters:**a - a file objectb - a file object **Returns:**true if the given file objects represent the same canonical file or zip file entry; false otherwise **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if either of the arguments were created with another file manager implementation

### getJavaFileObjectsFromFiles

[Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> **getJavaFileObjectsFromFiles**([Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> files)

Gets file objects representing the given files.

**Parameters:**files - a list of files **Returns:**a list of file objects **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the list of files includes a directory

### getJavaFileObjects

[Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> **getJavaFileObjects**([File](http://docs.google.com/java/io/File.html)... files)

Gets file objects representing the given files. Convenience method equivalent to:

getJavaFileObjectsFromFiles([Arrays.asList](http://docs.google.com/java/util/Arrays.html#asList(T...))(files))

**Parameters:**files - an array of files **Returns:**a list of file objects **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the array of files includes a directory [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given array contains null elements

### getJavaFileObjectsFromStrings

[Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> **getJavaFileObjectsFromStrings**([Iterable](http://docs.google.com/java/lang/Iterable.html)<[String](http://docs.google.com/java/lang/String.html)> names)

Gets file objects representing the given file names.

**Parameters:**names - a list of file names **Returns:**a list of file objects **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the list of file names includes a directory

### getJavaFileObjects

[Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [JavaFileObject](http://docs.google.com/javax/tools/JavaFileObject.html)> **getJavaFileObjects**([String](http://docs.google.com/java/lang/String.html)... names)

Gets file objects representing the given file names. Convenience method equivalent to:

getJavaFileObjectsFromStrings([Arrays.asList](http://docs.google.com/java/util/Arrays.html#asList(T...))(names))

**Parameters:**names - a list of file names **Returns:**a list of file objects **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the array of file names includes a directory [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given array contains null elements

### setLocation

void **setLocation**([JavaFileManager.Location](http://docs.google.com/javax/tools/JavaFileManager.Location.html) location,  
 [Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> path)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Associates the given path with the given location. Any previous value will be discarded.

**Parameters:**location - a locationpath - a list of files, if null use the default path for this location **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if location is an output location and path does not contain exactly one element [IOException](http://docs.google.com/java/io/IOException.html) - if location is an output location and path does not represent an existing directory**See Also:**[getLocation(javax.tools.JavaFileManager.Location)](http://docs.google.com/javax/tools/StandardJavaFileManager.html#getLocation(javax.tools.JavaFileManager.Location))

### getLocation

[Iterable](http://docs.google.com/java/lang/Iterable.html)<? extends [File](http://docs.google.com/java/io/File.html)> **getLocation**([JavaFileManager.Location](http://docs.google.com/javax/tools/JavaFileManager.Location.html) location)

Gets the path associated with the given location.

**Parameters:**location - a location **Returns:**a list of files or null if this location has no associated path**See Also:**[setLocation(javax.tools.JavaFileManager.Location, java.lang.Iterable)](http://docs.google.com/javax/tools/StandardJavaFileManager.html#setLocation(javax.tools.JavaFileManager.Location,%20java.lang.Iterable))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StandardJavaFileManager.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/javax/tools/SimpleJavaFileObject.html)   [**NEXT CLASS**](http://docs.google.com/javax/tools/StandardLocation.html) | [**FRAMES**](http://docs.google.com/index.html?javax/tools/StandardJavaFileManager.html)    [**NO FRAMES**](http://docs.google.com/StandardJavaFileManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#4d34og8) |

[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).