| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/System.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/lang/SuppressWarnings.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/Thread.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/System.html)    [**NO FRAMES**](http://docs.google.com/System.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#3dy6vkm) | CONSTR | [METHOD](#17dp8vu) |

## **java.lang**

Class System

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.lang.System**

public final class **System**extends [Object](http://docs.google.com/java/lang/Object.html)

The System class contains several useful class fields and methods. It cannot be instantiated.

Among the facilities provided by the System class are standard input, standard output, and error output streams; access to externally defined properties and environment variables; a means of loading files and libraries; and a utility method for quickly copying a portion of an array.

**Since:** JDK1.0

| **Field Summary** | |
| --- | --- |
| static [PrintStream](http://docs.google.com/java/io/PrintStream.html) | [**err**](http://docs.google.com/java/lang/System.html#err)            The "standard" error output stream. |
| static [InputStream](http://docs.google.com/java/io/InputStream.html) | [**in**](http://docs.google.com/java/lang/System.html#in)            The "standard" input stream. |
| static [PrintStream](http://docs.google.com/java/io/PrintStream.html) | [**out**](http://docs.google.com/java/lang/System.html#out)            The "standard" output stream. |

| **Method Summary** | |
| --- | --- |
| static void | [**arraycopy**](http://docs.google.com/java/lang/System.html#arraycopy(java.lang.Object,%20int,%20java.lang.Object,%20int,%20int))([Object](http://docs.google.com/java/lang/Object.html) src, int srcPos, [Object](http://docs.google.com/java/lang/Object.html) dest, int destPos, int length)            Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array. |
| static [String](http://docs.google.com/java/lang/String.html) | [**clearProperty**](http://docs.google.com/java/lang/System.html#clearProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Removes the system property indicated by the specified key. |
| static [Console](http://docs.google.com/java/io/Console.html) | [**console**](http://docs.google.com/java/lang/System.html#console())()            Returns the unique [Console](http://docs.google.com/java/io/Console.html) object associated with the current Java virtual machine, if any. |
| static long | [**currentTimeMillis**](http://docs.google.com/java/lang/System.html#currentTimeMillis())()            Returns the current time in milliseconds. |
| static void | [**exit**](http://docs.google.com/java/lang/System.html#exit(int))(int status)            Terminates the currently running Java Virtual Machine. |
| static void | [**gc**](http://docs.google.com/java/lang/System.html#gc())()            Runs the garbage collector. |
| static [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[String](http://docs.google.com/java/lang/String.html)> | [**getenv**](http://docs.google.com/java/lang/System.html#getenv())()            Returns an unmodifiable string map view of the current system environment. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getenv**](http://docs.google.com/java/lang/System.html#getenv(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Gets the value of the specified environment variable. |
| static [Properties](http://docs.google.com/java/util/Properties.html) | [**getProperties**](http://docs.google.com/java/lang/System.html#getProperties())()            Determines the current system properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getProperty**](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Gets the system property indicated by the specified key. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getProperty**](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) key, [String](http://docs.google.com/java/lang/String.html) def)            Gets the system property indicated by the specified key. |
| static [SecurityManager](http://docs.google.com/java/lang/SecurityManager.html) | [**getSecurityManager**](http://docs.google.com/java/lang/System.html#getSecurityManager())()            Gets the system security interface. |
| static int | [**identityHashCode**](http://docs.google.com/java/lang/System.html#identityHashCode(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) x)            Returns the same hash code for the given object as would be returned by the default method hashCode(), whether or not the given object's class overrides hashCode(). |
| static [Channel](http://docs.google.com/java/nio/channels/Channel.html) | [**inheritedChannel**](http://docs.google.com/java/lang/System.html#inheritedChannel())()            Returns the channel inherited from the entity that created this Java virtual machine. |
| static void | [**load**](http://docs.google.com/java/lang/System.html#load(java.lang.String))([String](http://docs.google.com/java/lang/String.html) filename)            Loads a code file with the specified filename from the local file system as a dynamic library. |
| static void | [**loadLibrary**](http://docs.google.com/java/lang/System.html#loadLibrary(java.lang.String))([String](http://docs.google.com/java/lang/String.html) libname)            Loads the system library specified by the libname argument. |
| static [String](http://docs.google.com/java/lang/String.html) | [**mapLibraryName**](http://docs.google.com/java/lang/System.html#mapLibraryName(java.lang.String))([String](http://docs.google.com/java/lang/String.html) libname)            Maps a library name into a platform-specific string representing a native library. |
| static long | [**nanoTime**](http://docs.google.com/java/lang/System.html#nanoTime())()            Returns the current value of the most precise available system timer, in nanoseconds. |
| static void | [**runFinalization**](http://docs.google.com/java/lang/System.html#runFinalization())()            Runs the finalization methods of any objects pending finalization. |
| static void | [**runFinalizersOnExit**](http://docs.google.com/java/lang/System.html#runFinalizersOnExit(boolean))(boolean value)  **Deprecated.** *This method is inherently unsafe. It may result in finalizers being called on live objects while other threads are concurrently manipulating those objects, resulting in erratic behavior or deadlock.* |
| static void | [**setErr**](http://docs.google.com/java/lang/System.html#setErr(java.io.PrintStream))([PrintStream](http://docs.google.com/java/io/PrintStream.html) err)            Reassigns the "standard" error output stream. |
| static void | [**setIn**](http://docs.google.com/java/lang/System.html#setIn(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) in)            Reassigns the "standard" input stream. |
| static void | [**setOut**](http://docs.google.com/java/lang/System.html#setOut(java.io.PrintStream))([PrintStream](http://docs.google.com/java/io/PrintStream.html) out)            Reassigns the "standard" output stream. |
| static void | [**setProperties**](http://docs.google.com/java/lang/System.html#setProperties(java.util.Properties))([Properties](http://docs.google.com/java/util/Properties.html) props)            Sets the system properties to the Properties argument. |
| static [String](http://docs.google.com/java/lang/String.html) | [**setProperty**](http://docs.google.com/java/lang/System.html#setProperty(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) key, [String](http://docs.google.com/java/lang/String.html) value)            Sets the system property indicated by the specified key. |
| static void | [**setSecurityManager**](http://docs.google.com/java/lang/System.html#setSecurityManager(java.lang.SecurityManager))([SecurityManager](http://docs.google.com/java/lang/SecurityManager.html) s)            Sets the System security. |

| **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)) |

| **Field Detail** |
| --- |

### in

public static final [InputStream](http://docs.google.com/java/io/InputStream.html) **in**

The "standard" input stream. This stream is already open and ready to supply input data. Typically this stream corresponds to keyboard input or another input source specified by the host environment or user.

### out

public static final [PrintStream](http://docs.google.com/java/io/PrintStream.html) **out**

The "standard" output stream. This stream is already open and ready to accept output data. Typically this stream corresponds to display output or another output destination specified by the host environment or user.

For simple stand-alone Java applications, a typical way to write a line of output data is:

System.out.println(data)

See the println methods in class PrintStream.

**See Also:**[PrintStream.println()](http://docs.google.com/java/io/PrintStream.html#println()), [PrintStream.println(boolean)](http://docs.google.com/java/io/PrintStream.html#println(boolean)), [PrintStream.println(char)](http://docs.google.com/java/io/PrintStream.html#println(char)), [PrintStream.println(char[])](http://docs.google.com/java/io/PrintStream.html#println(char%5B%5D)), [PrintStream.println(double)](http://docs.google.com/java/io/PrintStream.html#println(double)), [PrintStream.println(float)](http://docs.google.com/java/io/PrintStream.html#println(float)), [PrintStream.println(int)](http://docs.google.com/java/io/PrintStream.html#println(int)), [PrintStream.println(long)](http://docs.google.com/java/io/PrintStream.html#println(long)), [PrintStream.println(java.lang.Object)](http://docs.google.com/java/io/PrintStream.html#println(java.lang.Object)), [PrintStream.println(java.lang.String)](http://docs.google.com/java/io/PrintStream.html#println(java.lang.String))

### err

public static final [PrintStream](http://docs.google.com/java/io/PrintStream.html) **err**

The "standard" error output stream. This stream is already open and ready to accept output data.

Typically this stream corresponds to display output or another output destination specified by the host environment or user. By convention, this output stream is used to display error messages or other information that should come to the immediate attention of a user even if the principal output stream, the value of the variable out, has been redirected to a file or other destination that is typically not continuously monitored.

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

### setIn

public static void **setIn**([InputStream](http://docs.google.com/java/io/InputStream.html) in)

Reassigns the "standard" input stream.

First, if there is a security manager, its checkPermission method is called with a RuntimePermission("setIO") permission to see if it's ok to reassign the "standard" input stream.

**Parameters:**in - the new standard input stream. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow reassigning of the standard input stream.**Since:** JDK1.1 **See Also:**[SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)

### setOut

public static void **setOut**([PrintStream](http://docs.google.com/java/io/PrintStream.html) out)

Reassigns the "standard" output stream.

First, if there is a security manager, its checkPermission method is called with a RuntimePermission("setIO") permission to see if it's ok to reassign the "standard" output stream.

**Parameters:**out - the new standard output stream **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow reassigning of the standard output stream.**Since:** JDK1.1 **See Also:**[SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)

### setErr

public static void **setErr**([PrintStream](http://docs.google.com/java/io/PrintStream.html) err)

Reassigns the "standard" error output stream.

First, if there is a security manager, its checkPermission method is called with a RuntimePermission("setIO") permission to see if it's ok to reassign the "standard" error output stream.

**Parameters:**err - the new standard error output stream. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow reassigning of the standard error output stream.**Since:** JDK1.1 **See Also:**[SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)

### console

public static [Console](http://docs.google.com/java/io/Console.html) **console**()

Returns the unique [Console](http://docs.google.com/java/io/Console.html) object associated with the current Java virtual machine, if any.

**Returns:**The system console, if any, otherwise null.**Since:** 1.6

### inheritedChannel

public static [Channel](http://docs.google.com/java/nio/channels/Channel.html) **inheritedChannel**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Returns the channel inherited from the entity that created this Java virtual machine.

This method returns the channel obtained by invoking the [inheritedChannel](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html#inheritedChannel()) method of the system-wide default [SelectorProvider](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html) object.

In addition to the network-oriented channels described in [inheritedChannel](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html#inheritedChannel()), this method may return other kinds of channels in the future.

**Returns:**The inherited channel, if any, otherwise null. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - If an I/O error occurs [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - If a security manager is present and it does not permit access to the channel.**Since:** 1.5

### setSecurityManager

public static void **setSecurityManager**([SecurityManager](http://docs.google.com/java/lang/SecurityManager.html) s)

Sets the System security.

If there is a security manager already installed, this method first calls the security manager's checkPermission method with a RuntimePermission("setSecurityManager") permission to ensure it's ok to replace the existing security manager. This may result in throwing a SecurityException.

Otherwise, the argument is established as the current security manager. If the argument is null and no security manager has been established, then no action is taken and the method simply returns.

**Parameters:**s - the security manager. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if the security manager has already been set and its checkPermission method doesn't allow it to be replaced.**See Also:**[getSecurityManager()](http://docs.google.com/java/lang/System.html#getSecurityManager()), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)), [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)

### getSecurityManager

public static [SecurityManager](http://docs.google.com/java/lang/SecurityManager.html) **getSecurityManager**()

Gets the system security interface.

**Returns:**if a security manager has already been established for the current application, then that security manager is returned; otherwise, null is returned.**See Also:**[setSecurityManager(java.lang.SecurityManager)](http://docs.google.com/java/lang/System.html#setSecurityManager(java.lang.SecurityManager))

### currentTimeMillis

public static long **currentTimeMillis**()

Returns the current time in milliseconds. Note that while the unit of time of the return value is a millisecond, the granularity of the value depends on the underlying operating system and may be larger. For example, many operating systems measure time in units of tens of milliseconds.

See the description of the class Date for a discussion of slight discrepancies that may arise between "computer time" and coordinated universal time (UTC).

**Returns:**the difference, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC.**See Also:**[Date](http://docs.google.com/java/util/Date.html)

### nanoTime

public static long **nanoTime**()

Returns the current value of the most precise available system timer, in nanoseconds.

This method can only be used to measure elapsed time and is not related to any other notion of system or wall-clock time. The value returned represents nanoseconds since some fixed but arbitrary time (perhaps in the future, so values may be negative). This method provides nanosecond precision, but not necessarily nanosecond accuracy. No guarantees are made about how frequently values change. Differences in successive calls that span greater than approximately 292 years (263 nanoseconds) will not accurately compute elapsed time due to numerical overflow.

For example, to measure how long some code takes to execute:

long startTime = System.nanoTime();  
 // ... the code being measured ...  
 long estimatedTime = System.nanoTime() - startTime;

**Returns:**The current value of the system timer, in nanoseconds.**Since:** 1.5

### arraycopy

public static void **arraycopy**([Object](http://docs.google.com/java/lang/Object.html) src,  
 int srcPos,  
 [Object](http://docs.google.com/java/lang/Object.html) dest,  
 int destPos,  
 int length)

Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array. A subsequence of array components are copied from the source array referenced by src to the destination array referenced by dest. The number of components copied is equal to the length argument. The components at positions srcPos through srcPos+length-1 in the source array are copied into positions destPos through destPos+length-1, respectively, of the destination array.

If the src and dest arguments refer to the same array object, then the copying is performed as if the components at positions srcPos through srcPos+length-1 were first copied to a temporary array with length components and then the contents of the temporary array were copied into positions destPos through destPos+length-1 of the destination array.

If dest is null, then a NullPointerException is thrown.

If src is null, then a NullPointerException is thrown and the destination array is not modified.

Otherwise, if any of the following is true, an ArrayStoreException is thrown and the destination is not modified:

* The src argument refers to an object that is not an array.
* The dest argument refers to an object that is not an array.
* The src argument and dest argument refer to arrays whose component types are different primitive types.
* The src argument refers to an array with a primitive component type and the dest argument refers to an array with a reference component type.
* The src argument refers to an array with a reference component type and the dest argument refers to an array with a primitive component type.

Otherwise, if any of the following is true, an IndexOutOfBoundsException is thrown and the destination is not modified:

* The srcPos argument is negative.
* The destPos argument is negative.
* The length argument is negative.
* srcPos+length is greater than src.length, the length of the source array.
* destPos+length is greater than dest.length, the length of the destination array.

Otherwise, if any actual component of the source array from position srcPos through srcPos+length-1 cannot be converted to the component type of the destination array by assignment conversion, an ArrayStoreException is thrown. In this case, let ***k*** be the smallest nonnegative integer less than length such that src[srcPos+*k*] cannot be converted to the component type of the destination array; when the exception is thrown, source array components from positions srcPos through srcPos+*k*-1 will already have been copied to destination array positions destPos through destPos+*k*-1 and no other positions of the destination array will have been modified. (Because of the restrictions already itemized, this paragraph effectively applies only to the situation where both arrays have component types that are reference types.)

**Parameters:**src - the source array.srcPos - starting position in the source array.dest - the destination array.destPos - starting position in the destination data.length - the number of array elements to be copied. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if copying would cause access of data outside array bounds. [ArrayStoreException](http://docs.google.com/java/lang/ArrayStoreException.html) - if an element in the src array could not be stored into the dest array because of a type mismatch. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if either src or dest is null.

### identityHashCode

public static int **identityHashCode**([Object](http://docs.google.com/java/lang/Object.html) x)

Returns the same hash code for the given object as would be returned by the default method hashCode(), whether or not the given object's class overrides hashCode(). The hash code for the null reference is zero.

**Parameters:**x - object for which the hashCode is to be calculated **Returns:**the hashCode**Since:** JDK1.1

### getProperties

public static [Properties](http://docs.google.com/java/util/Properties.html) **getProperties**()

Determines the current system properties.

First, if there is a security manager, its checkPropertiesAccess method is called with no arguments. This may result in a security exception.

The current set of system properties for use by the [getProperty(String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)) method is returned as a Properties object. If there is no current set of system properties, a set of system properties is first created and initialized. This set of system properties always includes values for the following keys:

| Key | Description of Associated Value |
| --- | --- |
| java.version | Java Runtime Environment version |
| java.vendor | Java Runtime Environment vendor |
| java.vendor.url | Java vendor URL |
| java.home | Java installation directory |
| java.vm.specification.version | Java Virtual Machine specification version |
| java.vm.specification.vendor | Java Virtual Machine specification vendor |
| java.vm.specification.name | Java Virtual Machine specification name |
| java.vm.version | Java Virtual Machine implementation version |
| java.vm.vendor | Java Virtual Machine implementation vendor |
| java.vm.name | Java Virtual Machine implementation name |
| java.specification.version | Java Runtime Environment specification version |
| java.specification.vendor | Java Runtime Environment specification vendor |
| java.specification.name | Java Runtime Environment specification name |
| java.class.version | Java class format version number |
| java.class.path | Java class path |
| java.library.path | List of paths to search when loading libraries |
| java.io.tmpdir | Default temp file path |
| java.compiler | Name of JIT compiler to use |
| java.ext.dirs | Path of extension directory or directories |
| os.name | Operating system name |
| os.arch | Operating system architecture |
| os.version | Operating system version |
| file.separator | File separator ("/" on UNIX) |
| path.separator | Path separator (":" on UNIX) |
| line.separator | Line separator ("\n" on UNIX) |
| user.name | User's account name |
| user.home | User's home directory |
| user.dir | User's current working directory |

Multiple paths in a system property value are separated by the path separator character of the platform.

Note that even if the security manager does not permit the getProperties operation, it may choose to permit the [getProperty(String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)) operation.

**Returns:**the system properties **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPropertiesAccess method doesn't allow access to the system properties.**See Also:**[setProperties(java.util.Properties)](http://docs.google.com/java/lang/System.html#setProperties(java.util.Properties)), [SecurityException](http://docs.google.com/java/lang/SecurityException.html), [SecurityManager.checkPropertiesAccess()](http://docs.google.com/java/lang/SecurityManager.html#checkPropertiesAccess()), [Properties](http://docs.google.com/java/util/Properties.html)

### setProperties

public static void **setProperties**([Properties](http://docs.google.com/java/util/Properties.html) props)

Sets the system properties to the Properties argument.

First, if there is a security manager, its checkPropertiesAccess method is called with no arguments. This may result in a security exception.

The argument becomes the current set of system properties for use by the [getProperty(String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)) method. If the argument is null, then the current set of system properties is forgotten.

**Parameters:**props - the new system properties. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPropertiesAccess method doesn't allow access to the system properties.**See Also:**[getProperties()](http://docs.google.com/java/lang/System.html#getProperties()), [Properties](http://docs.google.com/java/util/Properties.html), [SecurityException](http://docs.google.com/java/lang/SecurityException.html), [SecurityManager.checkPropertiesAccess()](http://docs.google.com/java/lang/SecurityManager.html#checkPropertiesAccess())

### getProperty

public static [String](http://docs.google.com/java/lang/String.html) **getProperty**([String](http://docs.google.com/java/lang/String.html) key)

Gets the system property indicated by the specified key.

First, if there is a security manager, its checkPropertyAccess method is called with the key as its argument. This may result in a SecurityException.

If there is no current set of system properties, a set of system properties is first created and initialized in the same manner as for the getProperties method.

**Parameters:**key - the name of the system property. **Returns:**the string value of the system property, or null if there is no property with that key. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPropertyAccess method doesn't allow access to the specified system property. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is empty.**See Also:**[setProperty(java.lang.String, java.lang.String)](http://docs.google.com/java/lang/System.html#setProperty(java.lang.String,%20java.lang.String)), [SecurityException](http://docs.google.com/java/lang/SecurityException.html), [SecurityManager.checkPropertyAccess(java.lang.String)](http://docs.google.com/java/lang/SecurityManager.html#checkPropertyAccess(java.lang.String)), [getProperties()](http://docs.google.com/java/lang/System.html#getProperties())

### getProperty

public static [String](http://docs.google.com/java/lang/String.html) **getProperty**([String](http://docs.google.com/java/lang/String.html) key,  
 [String](http://docs.google.com/java/lang/String.html) def)

Gets the system property indicated by the specified key.

First, if there is a security manager, its checkPropertyAccess method is called with the key as its argument.

If there is no current set of system properties, a set of system properties is first created and initialized in the same manner as for the getProperties method.

**Parameters:**key - the name of the system property.def - a default value. **Returns:**the string value of the system property, or the default value if there is no property with that key. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPropertyAccess method doesn't allow access to the specified system property. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is empty.**See Also:**[setProperty(java.lang.String, java.lang.String)](http://docs.google.com/java/lang/System.html#setProperty(java.lang.String,%20java.lang.String)), [SecurityManager.checkPropertyAccess(java.lang.String)](http://docs.google.com/java/lang/SecurityManager.html#checkPropertyAccess(java.lang.String)), [getProperties()](http://docs.google.com/java/lang/System.html#getProperties())

### setProperty

public static [String](http://docs.google.com/java/lang/String.html) **setProperty**([String](http://docs.google.com/java/lang/String.html) key,  
 [String](http://docs.google.com/java/lang/String.html) value)

Sets the system property indicated by the specified key.

First, if a security manager exists, its SecurityManager.checkPermission method is called with a PropertyPermission(key, "write") permission. This may result in a SecurityException being thrown. If no exception is thrown, the specified property is set to the given value.

**Parameters:**key - the name of the system property.value - the value of the system property. **Returns:**the previous value of the system property, or null if it did not have one. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPermission method doesn't allow setting of the specified property. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key or value is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is empty.**Since:** 1.2 **See Also:**[getProperty(java.lang.String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)), [getProperty(java.lang.String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)), [getProperty(java.lang.String, java.lang.String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String,%20java.lang.String)), [PropertyPermission](http://docs.google.com/java/util/PropertyPermission.html), [SecurityManager.checkPermission(java.security.Permission)](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission))

### clearProperty

public static [String](http://docs.google.com/java/lang/String.html) **clearProperty**([String](http://docs.google.com/java/lang/String.html) key)

Removes the system property indicated by the specified key.

First, if a security manager exists, its SecurityManager.checkPermission method is called with a PropertyPermission(key, "write") permission. This may result in a SecurityException being thrown. If no exception is thrown, the specified property is removed.

**Parameters:**key - the name of the system property to be removed. **Returns:**the previous string value of the system property, or null if there was no property with that key. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkPropertyAccess method doesn't allow access to the specified system property. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is empty.**Since:** 1.5 **See Also:**[getProperty(java.lang.String)](http://docs.google.com/java/lang/System.html#getProperty(java.lang.String)), [setProperty(java.lang.String, java.lang.String)](http://docs.google.com/java/lang/System.html#setProperty(java.lang.String,%20java.lang.String)), [Properties](http://docs.google.com/java/util/Properties.html), [SecurityException](http://docs.google.com/java/lang/SecurityException.html), [SecurityManager.checkPropertiesAccess()](http://docs.google.com/java/lang/SecurityManager.html#checkPropertiesAccess())

### getenv

public static [String](http://docs.google.com/java/lang/String.html) **getenv**([String](http://docs.google.com/java/lang/String.html) name)

Gets the value of the specified environment variable. An environment variable is a system-dependent external named value.

If a security manager exists, its [checkPermission](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)) method is called with a [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)("getenv."+name) permission. This may result in a [SecurityException](http://docs.google.com/java/lang/SecurityException.html) being thrown. If no exception is thrown the value of the variable name is returned.

*System properties* and *environment variables* are both conceptually mappings between names and values. Both mechanisms can be used to pass user-defined information to a Java process. Environment variables have a more global effect, because they are visible to all descendants of the process which defines them, not just the immediate Java subprocess. They can have subtly different semantics, such as case insensitivity, on different operating systems. For these reasons, environment variables are more likely to have unintended side effects. It is best to use system properties where possible. Environment variables should be used when a global effect is desired, or when an external system interface requires an environment variable (such as PATH).

On UNIX systems the alphabetic case of name is typically significant, while on Microsoft Windows systems it is typically not. For example, the expression System.getenv("FOO").equals(System.getenv("foo")) is likely to be true on Microsoft Windows.

**Parameters:**name - the name of the environment variable **Returns:**the string value of the variable, or null if the variable is not defined in the system environment **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if name is null [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its [checkPermission](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)) method doesn't allow access to the environment variable name**See Also:**[getenv()](http://docs.google.com/java/lang/System.html#getenv()), [ProcessBuilder.environment()](http://docs.google.com/java/lang/ProcessBuilder.html#environment())

### getenv

public static [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[String](http://docs.google.com/java/lang/String.html)> **getenv**()

Returns an unmodifiable string map view of the current system environment. The environment is a system-dependent mapping from names to values which is passed from parent to child processes.

If the system does not support environment variables, an empty map is returned.

The returned map will never contain null keys or values. Attempting to query the presence of a null key or value will throw a [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html). Attempting to query the presence of a key or value which is not of type [String](http://docs.google.com/java/lang/String.html) will throw a [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html).

The returned map and its collection views may not obey the general contract of the [Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) and [Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()) methods.

The returned map is typically case-sensitive on all platforms.

If a security manager exists, its [checkPermission](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)) method is called with a [RuntimePermission](http://docs.google.com/java/lang/RuntimePermission.html)("getenv.\*") permission. This may result in a [SecurityException](http://docs.google.com/java/lang/SecurityException.html) being thrown.

When passing information to a Java subprocess, [system properties](#49x2ik5) are generally preferred over environment variables.

**Returns:**the environment as a map of variable names to values **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its [checkPermission](http://docs.google.com/java/lang/SecurityManager.html#checkPermission(java.security.Permission)) method doesn't allow access to the process environment**Since:** 1.5 **See Also:**[getenv(String)](http://docs.google.com/java/lang/System.html#getenv(java.lang.String)), [ProcessBuilder.environment()](http://docs.google.com/java/lang/ProcessBuilder.html#environment())

### exit

public static void **exit**(int status)

Terminates the currently running Java Virtual Machine. The argument serves as a status code; by convention, a nonzero status code indicates abnormal termination.

This method calls the exit method in class Runtime. This method never returns normally.

The call System.exit(n) is effectively equivalent to the call:

Runtime.getRuntime().exit(n)

**Parameters:**status - exit status. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkExit method doesn't allow exit with the specified status.**See Also:**[Runtime.exit(int)](http://docs.google.com/java/lang/Runtime.html#exit(int))

### gc

public static void **gc**()

Runs the garbage collector.

Calling the gc method suggests that the Java Virtual Machine expend effort toward recycling unused objects in order to make the memory they currently occupy available for quick reuse. When control returns from the method call, the Java Virtual Machine has made a best effort to reclaim space from all discarded objects.

The call System.gc() is effectively equivalent to the call:

Runtime.getRuntime().gc()

**See Also:**[Runtime.gc()](http://docs.google.com/java/lang/Runtime.html#gc())

### runFinalization

public static void **runFinalization**()

Runs the finalization methods of any objects pending finalization.

Calling this method suggests that the Java Virtual Machine expend effort toward running the finalize methods of objects that have been found to be discarded but whose finalize methods have not yet been run. When control returns from the method call, the Java Virtual Machine has made a best effort to complete all outstanding finalizations.

The call System.runFinalization() is effectively equivalent to the call:

Runtime.getRuntime().runFinalization()

**See Also:**[Runtime.runFinalization()](http://docs.google.com/java/lang/Runtime.html#runFinalization())

### runFinalizersOnExit

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public static void **runFinalizersOnExit**(boolean value)

**Deprecated.** *This method is inherently unsafe. It may result in finalizers being called on live objects while other threads are concurrently manipulating those objects, resulting in erratic behavior or deadlock.*

Enable or disable finalization on exit; doing so specifies that the finalizers of all objects that have finalizers that have not yet been automatically invoked are to be run before the Java runtime exits. By default, finalization on exit is disabled.

If there is a security manager, its checkExit method is first called with 0 as its argument to ensure the exit is allowed. This could result in a SecurityException.

**Parameters:**value - indicating enabling or disabling of finalization **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkExit method doesn't allow the exit.**Since:** JDK1.1 **See Also:**[Runtime.exit(int)](http://docs.google.com/java/lang/Runtime.html#exit(int)), [Runtime.gc()](http://docs.google.com/java/lang/Runtime.html#gc()), [SecurityManager.checkExit(int)](http://docs.google.com/java/lang/SecurityManager.html#checkExit(int))

### load

public static void **load**([String](http://docs.google.com/java/lang/String.html) filename)

Loads a code file with the specified filename from the local file system as a dynamic library. The filename argument must be a complete path name.

The call System.load(name) is effectively equivalent to the call:

Runtime.getRuntime().load(name)

**Parameters:**filename - the file to load. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkLink method doesn't allow loading of the specified dynamic library [UnsatisfiedLinkError](http://docs.google.com/java/lang/UnsatisfiedLinkError.html) - if the file does not exist. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if filename is null**See Also:**[Runtime.load(java.lang.String)](http://docs.google.com/java/lang/Runtime.html#load(java.lang.String)), [SecurityManager.checkLink(java.lang.String)](http://docs.google.com/java/lang/SecurityManager.html#checkLink(java.lang.String))

### loadLibrary

public static void **loadLibrary**([String](http://docs.google.com/java/lang/String.html) libname)

Loads the system library specified by the libname argument. The manner in which a library name is mapped to the actual system library is system dependent.

The call System.loadLibrary(name) is effectively equivalent to the call

Runtime.getRuntime().loadLibrary(name)

**Parameters:**libname - the name of the library. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if a security manager exists and its checkLink method doesn't allow loading of the specified dynamic library [UnsatisfiedLinkError](http://docs.google.com/java/lang/UnsatisfiedLinkError.html) - if the library does not exist. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if libname is null**See Also:**[Runtime.loadLibrary(java.lang.String)](http://docs.google.com/java/lang/Runtime.html#loadLibrary(java.lang.String)), [SecurityManager.checkLink(java.lang.String)](http://docs.google.com/java/lang/SecurityManager.html#checkLink(java.lang.String))

### mapLibraryName

public static [String](http://docs.google.com/java/lang/String.html) **mapLibraryName**([String](http://docs.google.com/java/lang/String.html) libname)

Maps a library name into a platform-specific string representing a native library.

**Parameters:**libname - the name of the library. **Returns:**a platform-dependent native library name. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if libname is null**Since:** 1.2 **See Also:**[loadLibrary(java.lang.String)](http://docs.google.com/java/lang/System.html#loadLibrary(java.lang.String)), [ClassLoader.findLibrary(java.lang.String)](http://docs.google.com/java/lang/ClassLoader.html#findLibrary(java.lang.String))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/System.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/lang/SuppressWarnings.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/Thread.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/System.html)    [**NO FRAMES**](http://docs.google.com/System.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#3dy6vkm) | CONSTR | [METHOD](#17dp8vu) |

[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).