| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | [**Class**](http://docs.google.com/java/lang/IllegalStateException.html) | **Use** | [**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   NEXT | [**FRAMES**](http://docs.google.com/index.html?java/lang//class-useIllegalStateException.html)    [**NO FRAMES**](http://docs.google.com/IllegalStateException.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

**Uses of Class**

**java.lang.IllegalStateException**

| Packages that use [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) | |
| --- | --- |
| [**java.awt**](#3znysh7) | Contains all of the classes for creating user interfaces and for painting graphics and images. |
| [**java.awt.dnd**](#2et92p0) | Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI. |
| [**java.nio**](#tyjcwt) | Defines buffers, which are containers for data, and provides an overview of the other NIO packages. |
| [**java.nio.channels**](#3dy6vkm) | Defines channels, which represent connections to entities that are capable of performing I/O operations, such as files and sockets; defines selectors, for multiplexed, non-blocking I/O operations. |
| [**java.util**](#1t3h5sf) | Contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (a string tokenizer, a random-number generator, and a bit array). |
| [**java.util.concurrent**](#4d34og8) | Utility classes commonly useful in concurrent programming. |
| [**javax.crypto**](#2s8eyo1) | Provides the classes and interfaces for cryptographic operations. |
| [**javax.swing**](#17dp8vu) | Provides a set of "lightweight" (all-Java language) components that, to the maximum degree possible, work the same on all platforms. |
| [**javax.xml.bind**](#3rdcrjn) | Provides a runtime binding framework for client applications including unmarshalling, marshalling, and validation capabilities. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.awt](http://docs.google.com/java/awt/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.awt](http://docs.google.com/java/awt/package-summary.html) | |
| --- | --- |
| class | [**IllegalComponentStateException**](http://docs.google.com/java/awt/IllegalComponentStateException.html)            Signals that an AWT component is not in an appropriate state for the requested operation. |

| Methods in [java.awt](http://docs.google.com/java/awt/package-summary.html) that throw [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) | |
| --- | --- |
| void | **SplashScreen.**[**close**](http://docs.google.com/java/awt/SplashScreen.html#close())()            Hides the splash screen, closes the window, and releases all associated resources. |
| [Graphics2D](http://docs.google.com/java/awt/Graphics2D.html) | **SplashScreen.**[**createGraphics**](http://docs.google.com/java/awt/SplashScreen.html#createGraphics())()            Creates a graphics context (as a [Graphics2D](http://docs.google.com/java/awt/Graphics2D.html) object) for the splash screen overlay image, which allows you to draw over the splash screen. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | **SplashScreen.**[**getBounds**](http://docs.google.com/java/awt/SplashScreen.html#getBounds())()            Returns the bounds of the splash screen window as a [Rectangle](http://docs.google.com/java/awt/Rectangle.html). |
| [URL](http://docs.google.com/java/net/URL.html) | **SplashScreen.**[**getImageURL**](http://docs.google.com/java/awt/SplashScreen.html#getImageURL())()            Returns the current splash screen image. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | **SplashScreen.**[**getSize**](http://docs.google.com/java/awt/SplashScreen.html#getSize())()            Returns the size of the splash screen window as a [Dimension](http://docs.google.com/java/awt/Dimension.html). |
| void | **SplashScreen.**[**setImageURL**](http://docs.google.com/java/awt/SplashScreen.html#setImageURL(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) imageURL)            Changes the splash screen image. |
| void | **SplashScreen.**[**update**](http://docs.google.com/java/awt/SplashScreen.html#update())()            Updates the splash window with current contents of the overlay image. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.awt.dnd](http://docs.google.com/java/awt/dnd/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.awt.dnd](http://docs.google.com/java/awt/dnd/package-summary.html) | |
| --- | --- |
| class | [**InvalidDnDOperationException**](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)            This exception is thrown by various methods in the java.awt.dnd package. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.nio](http://docs.google.com/java/nio/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.nio](http://docs.google.com/java/nio/package-summary.html) | |
| --- | --- |
| class | [**InvalidMarkException**](http://docs.google.com/java/nio/InvalidMarkException.html)            Unchecked exception thrown when an attempt is made to reset a buffer when its mark is not defined. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.nio.channels](http://docs.google.com/java/nio/channels/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.nio.channels](http://docs.google.com/java/nio/channels/package-summary.html) | |
| --- | --- |
| class | [**AlreadyConnectedException**](http://docs.google.com/java/nio/channels/AlreadyConnectedException.html)            Unchecked exception thrown when an attempt is made to connect a [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) that is already connected. |
| class | [**CancelledKeyException**](http://docs.google.com/java/nio/channels/CancelledKeyException.html)            Unchecked exception thrown when an attempt is made to use a selection key that is no longer valid. |
| class | [**ClosedSelectorException**](http://docs.google.com/java/nio/channels/ClosedSelectorException.html)            Unchecked exception thrown when an attempt is made to invoke an I/O operation upon a closed selector. |
| class | [**ConnectionPendingException**](http://docs.google.com/java/nio/channels/ConnectionPendingException.html)            Unchecked exception thrown when an attempt is made to connect a [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) for which a non-blocking connection operation is already in progress. |
| class | [**IllegalBlockingModeException**](http://docs.google.com/java/nio/channels/IllegalBlockingModeException.html)            Unchecked exception thrown when a blocking-mode-specific operation is invoked upon a channel in the incorrect blocking mode. |
| class | [**NoConnectionPendingException**](http://docs.google.com/java/nio/channels/NoConnectionPendingException.html)            Unchecked exception thrown when the [finishConnect](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect()) method of a [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) is invoked without first successfully invoking its [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) method. |
| class | [**NonReadableChannelException**](http://docs.google.com/java/nio/channels/NonReadableChannelException.html)            Unchecked exception thrown when an attempt is made to read from a channel that was not originally opened for reading. |
| class | [**NonWritableChannelException**](http://docs.google.com/java/nio/channels/NonWritableChannelException.html)            Unchecked exception thrown when an attempt is made to write to a channel that was not originally opened for writing. |
| class | [**NotYetBoundException**](http://docs.google.com/java/nio/channels/NotYetBoundException.html)            Unchecked exception thrown when an attempt is made to invoke an I/O operation upon a server socket channel that is not yet bound. |
| class | [**NotYetConnectedException**](http://docs.google.com/java/nio/channels/NotYetConnectedException.html)            Unchecked exception thrown when an attempt is made to invoke an I/O operation upon a socket channel that is not yet connected. |
| class | [**OverlappingFileLockException**](http://docs.google.com/java/nio/channels/OverlappingFileLockException.html)            Unchecked exception thrown when an attempt is made to acquire a lock on a region of a file that overlaps a region already locked by the same Java virtual machine, or when another thread is already waiting to lock an overlapping region of the same file. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.util](http://docs.google.com/java/util/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.util](http://docs.google.com/java/util/package-summary.html) | |
| --- | --- |
| class | [**FormatterClosedException**](http://docs.google.com/java/util/FormatterClosedException.html)            Unchecked exception thrown when the formatter has been closed. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.util.concurrent](http://docs.google.com/java/util/concurrent/package-summary.html) | |
| --- | --- |

| Subclasses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [java.util.concurrent](http://docs.google.com/java/util/concurrent/package-summary.html) | |
| --- | --- |
| class | [**CancellationException**](http://docs.google.com/java/util/concurrent/CancellationException.html)            Exception indicating that the result of a value-producing task, such as a [FutureTask](http://docs.google.com/java/util/concurrent/FutureTask.html), cannot be retrieved because the task was cancelled. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [javax.crypto](http://docs.google.com/javax/crypto/package-summary.html) | |
| --- | --- |

| Methods in [javax.crypto](http://docs.google.com/javax/crypto/package-summary.html) that throw [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) | |
| --- | --- |
| byte[] | **Mac.**[**doFinal**](http://docs.google.com/javax/crypto/Mac.html#doFinal())()            Finishes the MAC operation. |
| byte[] | **Mac.**[**doFinal**](http://docs.google.com/javax/crypto/Mac.html#doFinal(byte%5B%5D))(byte[] input)            Processes the given array of bytes and finishes the MAC operation. |
| void | **Mac.**[**doFinal**](http://docs.google.com/javax/crypto/Mac.html#doFinal(byte%5B%5D,%20int))(byte[] output, int outOffset)            Finishes the MAC operation. |
| [Key](http://docs.google.com/java/security/Key.html) | **KeyAgreement.**[**doPhase**](http://docs.google.com/javax/crypto/KeyAgreement.html#doPhase(java.security.Key,%20boolean))([Key](http://docs.google.com/java/security/Key.html) key, boolean lastPhase)            Executes the next phase of this key agreement with the given key that was received from one of the other parties involved in this key agreement. |
| protected abstract  [Key](http://docs.google.com/java/security/Key.html) | **KeyAgreementSpi.**[**engineDoPhase**](http://docs.google.com/javax/crypto/KeyAgreementSpi.html#engineDoPhase(java.security.Key,%20boolean))([Key](http://docs.google.com/java/security/Key.html) key, boolean lastPhase)            Executes the next phase of this key agreement with the given key that was received from one of the other parties involved in this key agreement. |
| protected abstract  byte[] | **KeyAgreementSpi.**[**engineGenerateSecret**](http://docs.google.com/javax/crypto/KeyAgreementSpi.html#engineGenerateSecret())()            Generates the shared secret and returns it in a new buffer. |
| protected abstract  int | **KeyAgreementSpi.**[**engineGenerateSecret**](http://docs.google.com/javax/crypto/KeyAgreementSpi.html#engineGenerateSecret(byte%5B%5D,%20int))(byte[] sharedSecret, int offset)            Generates the shared secret, and places it into the buffer sharedSecret, beginning at offset inclusive. |
| protected abstract  [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html) | **KeyAgreementSpi.**[**engineGenerateSecret**](http://docs.google.com/javax/crypto/KeyAgreementSpi.html#engineGenerateSecret(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Creates the shared secret and returns it as a secret key object of the requested algorithm type. |
| byte[] | **KeyAgreement.**[**generateSecret**](http://docs.google.com/javax/crypto/KeyAgreement.html#generateSecret())()            Generates the shared secret and returns it in a new buffer. |
| int | **KeyAgreement.**[**generateSecret**](http://docs.google.com/javax/crypto/KeyAgreement.html#generateSecret(byte%5B%5D,%20int))(byte[] sharedSecret, int offset)            Generates the shared secret, and places it into the buffer sharedSecret, beginning at offset inclusive. |
| [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html) | **KeyAgreement.**[**generateSecret**](http://docs.google.com/javax/crypto/KeyAgreement.html#generateSecret(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Creates the shared secret and returns it as a SecretKey object of the specified algorithm. |
| byte[] | **ExemptionMechanism.**[**genExemptionBlob**](http://docs.google.com/javax/crypto/ExemptionMechanism.html#genExemptionBlob())()            Generates the exemption mechanism key blob. |
| int | **ExemptionMechanism.**[**genExemptionBlob**](http://docs.google.com/javax/crypto/ExemptionMechanism.html#genExemptionBlob(byte%5B%5D))(byte[] output)            Generates the exemption mechanism key blob, and stores the result in the output buffer. |
| int | **ExemptionMechanism.**[**genExemptionBlob**](http://docs.google.com/javax/crypto/ExemptionMechanism.html#genExemptionBlob(byte%5B%5D,%20int))(byte[] output, int outputOffset)            Generates the exemption mechanism key blob, and stores the result in the output buffer, starting at outputOffset inclusive. |
| int | **ExemptionMechanism.**[**getOutputSize**](http://docs.google.com/javax/crypto/ExemptionMechanism.html#getOutputSize(int))(int inputLen)            Returns the length in bytes that an output buffer would need to be in order to hold the result of the next [genExemptionBlob](http://docs.google.com/javax/crypto/ExemptionMechanism.html#genExemptionBlob(byte%5B%5D)) operation, given the input length inputLen (in bytes). |
| void | **Mac.**[**update**](http://docs.google.com/javax/crypto/Mac.html#update(byte))(byte input)            Processes the given byte. |
| void | **Mac.**[**update**](http://docs.google.com/javax/crypto/Mac.html#update(byte%5B%5D))(byte[] input)            Processes the given array of bytes. |
| void | **Mac.**[**update**](http://docs.google.com/javax/crypto/Mac.html#update(byte%5B%5D,%20int,%20int))(byte[] input, int offset, int len)            Processes the first len bytes in input, starting at offset inclusive. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [javax.swing](http://docs.google.com/javax/swing/package-summary.html) | |
| --- | --- |

| Methods in [javax.swing](http://docs.google.com/javax/swing/package-summary.html) that throw [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) | |
| --- | --- |
| void | **TransferHandler.**[**exportToClipboard**](http://docs.google.com/javax/swing/TransferHandler.html#exportToClipboard(javax.swing.JComponent,%20java.awt.datatransfer.Clipboard,%20int))([JComponent](http://docs.google.com/javax/swing/JComponent.html) comp, [Clipboard](http://docs.google.com/java/awt/datatransfer/Clipboard.html) clip, int action)            Causes a transfer from the given component to the given clipboard. |

| Uses of [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) in [javax.xml.bind](http://docs.google.com/javax/xml/bind/package-summary.html) | |
| --- | --- |

| Methods in [javax.xml.bind](http://docs.google.com/javax/xml/bind/package-summary.html) that throw [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | **UnmarshallerHandler.**[**getResult**](http://docs.google.com/javax/xml/bind/UnmarshallerHandler.html#getResult())()            Obtains the unmarshalled result. |

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | [**Class**](http://docs.google.com/java/lang/IllegalStateException.html) | **Use** | [**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   NEXT | [**FRAMES**](http://docs.google.com/index.html?java/lang//class-useIllegalStateException.html)    [**NO FRAMES**](http://docs.google.com/IllegalStateException.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[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).