| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Line.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/sound/sampled/FloatControl.Type.html)   [**NEXT CLASS**](http://docs.google.com/javax/sound/sampled/Line.Info.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sound/sampled/Line.html)    [**NO FRAMES**](http://docs.google.com/Line.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | CONSTR | [METHOD](#2et92p0) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

## **javax.sound.sampled**

Interface Line

**All Known Subinterfaces:** [Clip](http://docs.google.com/javax/sound/sampled/Clip.html), [DataLine](http://docs.google.com/javax/sound/sampled/DataLine.html), [Mixer](http://docs.google.com/javax/sound/sampled/Mixer.html), [Port](http://docs.google.com/javax/sound/sampled/Port.html), [SourceDataLine](http://docs.google.com/javax/sound/sampled/SourceDataLine.html), [TargetDataLine](http://docs.google.com/javax/sound/sampled/TargetDataLine.html)

public interface **Line**

The Line interface represents a mono or multi-channel audio feed. A line is an element of the digital audio "pipeline," such as a mixer, an input or output port, or a data path into or out of a mixer.

A line can have controls, such as gain, pan, and reverb. The controls themselves are instances of classes that extend the base [Control](http://docs.google.com/javax/sound/sampled/Control.html) class. The Line interface provides two accessor methods for obtaining the line's controls: [getControls](http://docs.google.com/javax/sound/sampled/Line.html#getControls()) returns the entire set, and [getControl](http://docs.google.com/javax/sound/sampled/Line.html#getControl(javax.sound.sampled.Control.Type)) returns a single control of specified type.

Lines exist in various states at different times. When a line opens, it reserves system resources for itself, and when it closes, these resources are freed for other objects or applications. The [isOpen()](http://docs.google.com/javax/sound/sampled/Line.html#isOpen()) method lets you discover whether a line is open or closed. An open line need not be processing data, however. Such processing is typically initiated by subinterface methods such as [SourceDataLine.write](http://docs.google.com/javax/sound/sampled/SourceDataLine.html#write(byte%5B%5D,%20int,%20int)) and [TargetDataLine.read](http://docs.google.com/javax/sound/sampled/TargetDataLine.html#read(byte%5B%5D,%20int,%20int)).

You can register an object to receive notifications whenever the line's state changes. The object must implement the [LineListener](http://docs.google.com/javax/sound/sampled/LineListener.html) interface, which consists of the single method [update](http://docs.google.com/javax/sound/sampled/LineListener.html#update(javax.sound.sampled.LineEvent)). This method will be invoked when a line opens and closes (and, if it's a [DataLine](http://docs.google.com/javax/sound/sampled/DataLine.html), when it starts and stops).

An object can be registered to listen to multiple lines. The event it receives in its update method will specify which line created the event, what type of event it was (OPEN, CLOSE, START, or STOP), and how many sample frames the line had processed at the time the event occurred.

Certain line operations, such as open and close, can generate security exceptions if invoked by unprivileged code when the line is a shared audio resource.

**Since:** 1.3 **See Also:**[LineEvent](http://docs.google.com/javax/sound/sampled/LineEvent.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**Line.Info**](http://docs.google.com/javax/sound/sampled/Line.Info.html)            A Line.Info object contains information about a line. |

| **Method Summary** | |
| --- | --- |
| void | [**addLineListener**](http://docs.google.com/javax/sound/sampled/Line.html#addLineListener(javax.sound.sampled.LineListener))([LineListener](http://docs.google.com/javax/sound/sampled/LineListener.html) listener)            Adds a listener to this line. |
| void | [**close**](http://docs.google.com/javax/sound/sampled/Line.html#close())()            Closes the line, indicating that any system resources in use by the line can be released. |
| [Control](http://docs.google.com/javax/sound/sampled/Control.html) | [**getControl**](http://docs.google.com/javax/sound/sampled/Line.html#getControl(javax.sound.sampled.Control.Type))([Control.Type](http://docs.google.com/javax/sound/sampled/Control.Type.html) control)            Obtains a control of the specified type, if there is any. |
| [Control](http://docs.google.com/javax/sound/sampled/Control.html)[] | [**getControls**](http://docs.google.com/javax/sound/sampled/Line.html#getControls())()            Obtains the set of controls associated with this line. |
| [Line.Info](http://docs.google.com/javax/sound/sampled/Line.Info.html) | [**getLineInfo**](http://docs.google.com/javax/sound/sampled/Line.html#getLineInfo())()            Obtains the Line.Info object describing this line. |
| boolean | [**isControlSupported**](http://docs.google.com/javax/sound/sampled/Line.html#isControlSupported(javax.sound.sampled.Control.Type))([Control.Type](http://docs.google.com/javax/sound/sampled/Control.Type.html) control)            Indicates whether the line supports a control of the specified type. |
| boolean | [**isOpen**](http://docs.google.com/javax/sound/sampled/Line.html#isOpen())()            Indicates whether the line is open, meaning that it has reserved system resources and is operational, although it might not currently be playing or capturing sound. |
| void | [**open**](http://docs.google.com/javax/sound/sampled/Line.html#open())()            Opens the line, indicating that it should acquire any required system resources and become operational. |
| void | [**removeLineListener**](http://docs.google.com/javax/sound/sampled/Line.html#removeLineListener(javax.sound.sampled.LineListener))([LineListener](http://docs.google.com/javax/sound/sampled/LineListener.html) listener)            Removes the specified listener from this line's list of listeners. |

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

### getLineInfo

[Line.Info](http://docs.google.com/javax/sound/sampled/Line.Info.html) **getLineInfo**()

Obtains the Line.Info object describing this line.

**Returns:**description of the line

### open

void **open**()  
 throws [LineUnavailableException](http://docs.google.com/javax/sound/sampled/LineUnavailableException.html)

Opens the line, indicating that it should acquire any required system resources and become operational. If this operation succeeds, the line is marked as open, and an OPEN event is dispatched to the line's listeners.

Note that some lines, once closed, cannot be reopened. Attempts to reopen such a line will always result in an LineUnavailableException.

Some types of lines have configurable properties that may affect resource allocation. For example, a DataLine must be opened with a particular format and buffer size. Such lines should provide a mechanism for configuring these properties, such as an additional open method or methods which allow an application to specify the desired settings.

This method takes no arguments, and opens the line with the current settings. For [SourceDataLine](http://docs.google.com/javax/sound/sampled/SourceDataLine.html) and [TargetDataLine](http://docs.google.com/javax/sound/sampled/TargetDataLine.html) objects, this means that the line is opened with default settings. For a [Clip](http://docs.google.com/javax/sound/sampled/Clip.html), however, the buffer size is determined when data is loaded. Since this method does not allow the application to specify any data to load, an IllegalArgumentException is thrown. Therefore, you should instead use one of the open methods provided in the Clip interface to load data into the Clip.

For DataLine's, if the DataLine.Info object which was used to retrieve the line, specifies at least one fully qualified audio format, the last one will be used as the default format.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this method is called on a Clip instance. [LineUnavailableException](http://docs.google.com/javax/sound/sampled/LineUnavailableException.html) - if the line cannot be opened due to resource restrictions. [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if the line cannot be opened due to security restrictions.**See Also:**[close()](http://docs.google.com/javax/sound/sampled/Line.html#close()), [isOpen()](http://docs.google.com/javax/sound/sampled/Line.html#isOpen()), [LineEvent](http://docs.google.com/javax/sound/sampled/LineEvent.html), [DataLine](http://docs.google.com/javax/sound/sampled/DataLine.html), [Clip.open(AudioFormat, byte[], int, int)](http://docs.google.com/javax/sound/sampled/Clip.html#open(javax.sound.sampled.AudioFormat,%20byte%5B%5D,%20int,%20int)), [Clip.open(AudioInputStream)](http://docs.google.com/javax/sound/sampled/Clip.html#open(javax.sound.sampled.AudioInputStream))

### close

void **close**()

Closes the line, indicating that any system resources in use by the line can be released. If this operation succeeds, the line is marked closed and a CLOSE event is dispatched to the line's listeners.

**Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if the line cannot be closed due to security restrictions.**See Also:**[open()](http://docs.google.com/javax/sound/sampled/Line.html#open()), [isOpen()](http://docs.google.com/javax/sound/sampled/Line.html#isOpen()), [LineEvent](http://docs.google.com/javax/sound/sampled/LineEvent.html)

### isOpen

boolean **isOpen**()

Indicates whether the line is open, meaning that it has reserved system resources and is operational, although it might not currently be playing or capturing sound.

**Returns:**true if the line is open, otherwise false**See Also:**[open()](http://docs.google.com/javax/sound/sampled/Line.html#open()), [close()](http://docs.google.com/javax/sound/sampled/Line.html#close())

### getControls

[Control](http://docs.google.com/javax/sound/sampled/Control.html)[] **getControls**()

Obtains the set of controls associated with this line. Some controls may only be available when the line is open. If there are no controls, this method returns an array of length 0.

**Returns:**the array of controls**See Also:**[getControl(javax.sound.sampled.Control.Type)](http://docs.google.com/javax/sound/sampled/Line.html#getControl(javax.sound.sampled.Control.Type))

### isControlSupported

boolean **isControlSupported**([Control.Type](http://docs.google.com/javax/sound/sampled/Control.Type.html) control)

Indicates whether the line supports a control of the specified type. Some controls may only be available when the line is open.

**Parameters:**control - the type of the control for which support is queried **Returns:**true if at least one control of the specified type is supported, otherwise false.

### getControl

[Control](http://docs.google.com/javax/sound/sampled/Control.html) **getControl**([Control.Type](http://docs.google.com/javax/sound/sampled/Control.Type.html) control)

Obtains a control of the specified type, if there is any. Some controls may only be available when the line is open.

**Parameters:**control - the type of the requested control **Returns:**a control of the specified type **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if a control of the specified type is not supported**See Also:**[getControls()](http://docs.google.com/javax/sound/sampled/Line.html#getControls()), [isControlSupported(Control.Type control)](http://docs.google.com/javax/sound/sampled/Line.html#isControlSupported(javax.sound.sampled.Control.Type))

### addLineListener

void **addLineListener**([LineListener](http://docs.google.com/javax/sound/sampled/LineListener.html) listener)

Adds a listener to this line. Whenever the line's status changes, the listener's update() method is called with a LineEvent object that describes the change.

**Parameters:**listener - the object to add as a listener to this line**See Also:**[removeLineListener(javax.sound.sampled.LineListener)](http://docs.google.com/javax/sound/sampled/Line.html#removeLineListener(javax.sound.sampled.LineListener)), [LineListener.update(javax.sound.sampled.LineEvent)](http://docs.google.com/javax/sound/sampled/LineListener.html#update(javax.sound.sampled.LineEvent)), [LineEvent](http://docs.google.com/javax/sound/sampled/LineEvent.html)

### removeLineListener

void **removeLineListener**([LineListener](http://docs.google.com/javax/sound/sampled/LineListener.html) listener)

Removes the specified listener from this line's list of listeners.

**Parameters:**listener - listener to remove**See Also:**[addLineListener(javax.sound.sampled.LineListener)](http://docs.google.com/javax/sound/sampled/Line.html#addLineListener(javax.sound.sampled.LineListener))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Line.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/sound/sampled/FloatControl.Type.html)   [**NEXT CLASS**](http://docs.google.com/javax/sound/sampled/Line.Info.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sound/sampled/Line.html)    [**NO FRAMES**](http://docs.google.com/Line.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | CONSTR | [METHOD](#2et92p0) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

[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).