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

## **javax.sound.midi**

Class MidiSystem

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.sound.midi.MidiSystem**

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

The MidiSystem class provides access to the installed MIDI system resources, including devices such as synthesizers, sequencers, and MIDI input and output ports. A typical simple MIDI application might begin by invoking one or more MidiSystem methods to learn what devices are installed and to obtain the ones needed in that application.

The class also has methods for reading files, streams, and URLs that contain standard MIDI file data or soundbanks. You can query the MidiSystem for the format of a specified MIDI file.

You cannot instantiate a MidiSystem; all the methods are static.

Properties can be used to specify default MIDI devices. Both system properties and a properties file are considered. The properties file is "lib/sound.properties" in the JRE directory. If a property exists both as a system property and in the properties file, the system property takes precedence. If none is specified, a suitable default is chosen among the available devices. The syntax of the properties file is specified in [Properties.load](http://docs.google.com/java/util/Properties.html#load(java.io.InputStream)). The following table lists the available property keys and which methods consider them:

| Property Key | Interface | Affected Method |
| --- | --- | --- |
| javax.sound.midi.Receiver | [Receiver](http://docs.google.com/javax/sound/midi/Receiver.html) | [getReceiver()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver()) |
| javax.sound.midi.Sequencer | [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) | [getSequencer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequencer()) |
| javax.sound.midi.Synthesizer | [Synthesizer](http://docs.google.com/javax/sound/midi/Synthesizer.html) | [getSynthesizer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer()) |
| javax.sound.midi.Transmitter | [Transmitter](http://docs.google.com/javax/sound/midi/Transmitter.html) | [getTransmitter()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getTransmitter()) |

The property value consists of the provider class name and the device name, separated by the hash mark ("#"). The provider class name is the fully-qualified name of a concrete [MIDI device provider](http://docs.google.com/javax/sound/midi/spi/MidiDeviceProvider.html) class. The device name is matched against the String returned by the getName method of MidiDevice.Info. Either the class name, or the device name may be omitted. If only the class name is specified, the trailing hash mark is optional.

If the provider class is specified, and it can be successully retrieved from the installed providers, the list of MidiDevice.Info objects is retrieved from the provider. Otherwise, or when these devices do not provide a subsequent match, the list is retrieved from [getMidiDeviceInfo()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiDeviceInfo()) to contain all available MidiDevice.Info objects.

If a device name is specified, the resulting list of MidiDevice.Info objects is searched: the first one with a matching name, and whose MidiDevice implements the respective interface, will be returned. If no matching MidiDevice.Info object is found, or the device name is not specified, the first suitable device from the resulting list will be returned. For Sequencer and Synthesizer, a device is suitable if it implements the respective interface; whereas for Receiver and Transmitter, a device is suitable if it implements neither Sequencer nor Synthesizer and provides at least one Receiver or Transmitter, respectively. For example, the property javax.sound.midi.Receiver with a value "com.sun.media.sound.MidiProvider#SunMIDI1" will have the following consequences when getReceiver is called: if the class com.sun.media.sound.MidiProvider exists in the list of installed MIDI device providers, the first Receiver device with name "SunMIDI1" will be returned. If it cannot be found, the first Receiver from that provider will be returned, regardless of name. If there is none, the first Receiver with name "SunMIDI1" in the list of all devices (as returned by getMidiDeviceInfo) will be returned, or, if not found, the first Receiver that can be found in the list of all devices is returned. If that fails, too, a MidiUnavailableException is thrown.

| **Method Summary** | |
| --- | --- |
| static [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html) | [**getMidiDevice**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiDevice(javax.sound.midi.MidiDevice.Info))([MidiDevice.Info](http://docs.google.com/javax/sound/midi/MidiDevice.Info.html) info)            Obtains the requested MIDI device. |
| static [MidiDevice.Info](http://docs.google.com/javax/sound/midi/MidiDevice.Info.html)[] | [**getMidiDeviceInfo**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiDeviceInfo())()            Obtains an array of information objects representing the set of all MIDI devices available on the system. |
| static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) | [**getMidiFileFormat**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.File))([File](http://docs.google.com/java/io/File.html) file)            Obtains the MIDI file format of the specified File. |
| static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) | [**getMidiFileFormat**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) stream)            Obtains the MIDI file format of the data in the specified input stream. |
| static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) | [**getMidiFileFormat**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Obtains the MIDI file format of the data in the specified URL. |
| static int[] | [**getMidiFileTypes**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileTypes())()            Obtains the set of MIDI file types for which file writing support is provided by the system. |
| static int[] | [**getMidiFileTypes**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileTypes(javax.sound.midi.Sequence))([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) sequence)            Obtains the set of MIDI file types that the system can write from the sequence specified. |
| static [Receiver](http://docs.google.com/javax/sound/midi/Receiver.html) | [**getReceiver**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver())()            Obtains a MIDI receiver from an external MIDI port or other default device. |
| static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) | [**getSequence**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequence(java.io.File))([File](http://docs.google.com/java/io/File.html) file)            Obtains a MIDI sequence from the specified File. |
| static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) | [**getSequence**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequence(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) stream)            Obtains a MIDI sequence from the specified input stream. |
| static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) | [**getSequence**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequence(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Obtains a MIDI sequence from the specified URL. |
| static [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) | [**getSequencer**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequencer())()            Obtains the default Sequencer, connected to a default device. |
| static [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) | [**getSequencer**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequencer(boolean))(boolean connected)            Obtains the default Sequencer, optionally connected to a default device. |
| static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) | [**getSoundbank**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSoundbank(java.io.File))([File](http://docs.google.com/java/io/File.html) file)            Constructs a Soundbank by reading it from the specified File. |
| static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) | [**getSoundbank**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSoundbank(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) stream)            Constructs a MIDI sound bank by reading it from the specified stream. |
| static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) | [**getSoundbank**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSoundbank(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Constructs a Soundbank by reading it from the specified URL. |
| static [Synthesizer](http://docs.google.com/javax/sound/midi/Synthesizer.html) | [**getSynthesizer**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer())()            Obtains the default synthesizer. |
| static [Transmitter](http://docs.google.com/javax/sound/midi/Transmitter.html) | [**getTransmitter**](http://docs.google.com/javax/sound/midi/MidiSystem.html#getTransmitter())()            Obtains a MIDI transmitter from an external MIDI port or other default source. |
| static boolean | [**isFileTypeSupported**](http://docs.google.com/javax/sound/midi/MidiSystem.html#isFileTypeSupported(int))(int fileType)            Indicates whether file writing support for the specified MIDI file type is provided by the system. |
| static boolean | [**isFileTypeSupported**](http://docs.google.com/javax/sound/midi/MidiSystem.html#isFileTypeSupported(int,%20javax.sound.midi.Sequence))(int fileType, [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) sequence)            Indicates whether a MIDI file of the file type specified can be written from the sequence indicated. |
| static int | [**write**](http://docs.google.com/javax/sound/midi/MidiSystem.html#write(javax.sound.midi.Sequence,%20int,%20java.io.File))([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) in, int type, [File](http://docs.google.com/java/io/File.html) out)            Writes a stream of bytes representing a file of the MIDI file type indicated to the external file provided. |
| static int | [**write**](http://docs.google.com/javax/sound/midi/MidiSystem.html#write(javax.sound.midi.Sequence,%20int,%20java.io.OutputStream))([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) in, int fileType, [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)            Writes a stream of bytes representing a file of the MIDI file type indicated to the output stream provided. |

| **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)) |

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

### getMidiDeviceInfo

public static [MidiDevice.Info](http://docs.google.com/javax/sound/midi/MidiDevice.Info.html)[] **getMidiDeviceInfo**()

Obtains an array of information objects representing the set of all MIDI devices available on the system. A returned information object can then be used to obtain the corresponding device object, by invoking [getMidiDevice](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiDevice(javax.sound.midi.MidiDevice.Info)).

**Returns:**an array of MidiDevice.Info objects, one for each installed MIDI device. If no such devices are installed, an array of length 0 is returned.

### getMidiDevice

public static [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html) **getMidiDevice**([MidiDevice.Info](http://docs.google.com/javax/sound/midi/MidiDevice.Info.html) info)  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains the requested MIDI device.

**Parameters:**info - a device information object representing the desired device. **Returns:**the requested device **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the requested device is not available due to resource restrictions [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the info object does not represent a MIDI device installed on the system**See Also:**[getMidiDeviceInfo()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiDeviceInfo())

### getReceiver

public static [Receiver](http://docs.google.com/javax/sound/midi/Receiver.html) **getReceiver**()  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains a MIDI receiver from an external MIDI port or other default device.

If the system property javax.sound.midi.Receiver is defined or it is defined in the file "sound.properties", it is used to identify the device that provides the default receiver. For details, refer to the [class description](http://docs.google.com/javax/sound/midi/MidiSystem.html). If a suitable MIDI port is not available, the Receiver is retrieved from an installed synthesizer.

If this method returns successfully, the [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html) the Receiver belongs to is opened implicitly, if it is not already open. It is possible to close an implicitly opened device by calling [close](http://docs.google.com/javax/sound/midi/Receiver.html#close()) on the returned Receiver. All open Receiver instances have to be closed in order to release system resources hold by the MidiDevice. For a detailed description of open/close behaviour see the class description of [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html).

**Returns:**the default MIDI receiver **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the default receiver is not available due to resource restrictions, or no device providing receivers is installed in the system

### getTransmitter

public static [Transmitter](http://docs.google.com/javax/sound/midi/Transmitter.html) **getTransmitter**()  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains a MIDI transmitter from an external MIDI port or other default source.

If the system property javax.sound.midi.Transmitter is defined or it is defined in the file "sound.properties", it is used to identify the device that provides the default transmitter. For details, refer to the [class description](http://docs.google.com/javax/sound/midi/MidiSystem.html). If this method returns successfully, the [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html) the Transmitter belongs to is opened implicitly, if it is not already open. It is possible to close an implicitly opened device by calling [close](http://docs.google.com/javax/sound/midi/Transmitter.html#close()) on the returned Transmitter. All open Transmitter instances have to be closed in order to release system resources hold by the MidiDevice. For a detailed description of open/close behaviour see the class description of [MidiDevice](http://docs.google.com/javax/sound/midi/MidiDevice.html).

**Returns:**the default MIDI transmitter **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the default transmitter is not available due to resource restrictions, or no device providing transmitters is installed in the system

### getSynthesizer

public static [Synthesizer](http://docs.google.com/javax/sound/midi/Synthesizer.html) **getSynthesizer**()  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains the default synthesizer.

If the system property javax.sound.midi.Synthesizer is defined or it is defined in the file "sound.properties", it is used to identify the default synthesizer. For details, refer to the [class description](http://docs.google.com/javax/sound/midi/MidiSystem.html).

**Returns:**the default synthesizer **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the synthesizer is not available due to resource restrictions, or no synthesizer is installed in the system

### getSequencer

public static [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) **getSequencer**()  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains the default Sequencer, connected to a default device. The returned Sequencer instance is connected to the default Synthesizer, as returned by [getSynthesizer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer()). If there is no Synthesizer available, or the default Synthesizer cannot be opened, the sequencer is connected to the default Receiver, as returned by [getReceiver()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver()). The connection is made by retrieving a Transmitter instance from the Sequencer and setting its Receiver. Closing and re-opening the sequencer will restore the connection to the default device.

This method is equivalent to calling getSequencer(true).

If the system property javax.sound.midi.Sequencer is defined or it is defined in the file "sound.properties", it is used to identify the default sequencer. For details, refer to the [class description](http://docs.google.com/javax/sound/midi/MidiSystem.html).

**Returns:**the default sequencer, connected to a default Receiver **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the sequencer is not available due to resource restrictions, or there is no Receiver available by any installed MidiDevice, or no sequencer is installed in the system.**See Also:**[getSequencer(boolean)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSequencer(boolean)), [getSynthesizer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer()), [getReceiver()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver())

### getSequencer

public static [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) **getSequencer**(boolean connected)  
 throws [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html)

Obtains the default Sequencer, optionally connected to a default device.

If connected is true, the returned Sequencer instance is connected to the default Synthesizer, as returned by [getSynthesizer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer()). If there is no Synthesizer available, or the default Synthesizer cannot be opened, the sequencer is connected to the default Receiver, as returned by [getReceiver()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver()). The connection is made by retrieving a Transmitter instance from the Sequencer and setting its Receiver. Closing and re-opening the sequencer will restore the connection to the default device.

If connected is false, the returned Sequencer instance is not connected, it has no open Transmitters. In order to play the sequencer on a MIDI device, or a Synthesizer, it is necessary to get a Transmitter and set its Receiver.

If the system property javax.sound.midi.Sequencer is defined or it is defined in the file "sound.properties", it is used to identify the default sequencer. For details, refer to the [class description](http://docs.google.com/javax/sound/midi/MidiSystem.html).

**Returns:**the default sequencer **Throws:** [MidiUnavailableException](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) - if the sequencer is not available due to resource restrictions, or no sequencer is installed in the system, or if connected is true, and there is no Receiver available by any installed MidiDevice**Since:** 1.5 **See Also:**[getSynthesizer()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getSynthesizer()), [getReceiver()](http://docs.google.com/javax/sound/midi/MidiSystem.html#getReceiver())

### getSoundbank

public static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) **getSoundbank**([InputStream](http://docs.google.com/java/io/InputStream.html) stream)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Constructs a MIDI sound bank by reading it from the specified stream. The stream must point to a valid MIDI soundbank file. In general, MIDI soundbank providers may need to read some data from the stream before determining whether they support it. These parsers must be able to mark the stream, read enough data to determine whether they support the stream, and, if not, reset the stream's read pointer to its original position. If the input stream does not support this, this method may fail with an IOException.

**Parameters:**stream - the source of the sound bank data. **Returns:**the sound bank **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the stream does not point to valid MIDI soundbank data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurred when loading the soundbank**See Also:**[InputStream.markSupported()](http://docs.google.com/java/io/InputStream.html#markSupported()), [InputStream.mark(int)](http://docs.google.com/java/io/InputStream.html#mark(int))

### getSoundbank

public static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) **getSoundbank**([URL](http://docs.google.com/java/net/URL.html) url)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Constructs a Soundbank by reading it from the specified URL. The URL must point to a valid MIDI soundbank file.

**Parameters:**url - the source of the sound bank data **Returns:**the sound bank **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the URL does not point to valid MIDI soundbank data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurred when loading the soundbank

### getSoundbank

public static [Soundbank](http://docs.google.com/javax/sound/midi/Soundbank.html) **getSoundbank**([File](http://docs.google.com/java/io/File.html) file)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Constructs a Soundbank by reading it from the specified File. The File must point to a valid MIDI soundbank file.

**Parameters:**file - the source of the sound bank data **Returns:**the sound bank **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the File does not point to valid MIDI soundbank data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurred when loading the soundbank

### getMidiFileFormat

public static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) **getMidiFileFormat**([InputStream](http://docs.google.com/java/io/InputStream.html) stream)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains the MIDI file format of the data in the specified input stream. The stream must point to valid MIDI file data for a file type recognized by the system.

This method and/or the code it invokes may need to read some data from the stream to determine whether its data format is supported. The implementation may therefore need to mark the stream, read enough data to determine whether it is in a supported format, and reset the stream's read pointer to its original position. If the input stream does not permit this set of operations, this method may fail with an IOException.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while determining the file format.

**Parameters:**stream - the input stream from which file format information should be extracted **Returns:**an MidiFileFormat object describing the MIDI file format **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the stream does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs while accessing the stream**See Also:**[getMidiFileFormat(URL)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.net.URL)), [getMidiFileFormat(File)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.File)), [InputStream.markSupported()](http://docs.google.com/java/io/InputStream.html#markSupported()), [InputStream.mark(int)](http://docs.google.com/java/io/InputStream.html#mark(int))

### getMidiFileFormat

public static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) **getMidiFileFormat**([URL](http://docs.google.com/java/net/URL.html) url)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains the MIDI file format of the data in the specified URL. The URL must point to valid MIDI file data for a file type recognized by the system.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while determining the file format.

**Parameters:**url - the URL from which file format information should be extracted **Returns:**a MidiFileFormat object describing the MIDI file format **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the URL does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs while accessing the URL**See Also:**[getMidiFileFormat(InputStream)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.InputStream)), [getMidiFileFormat(File)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.File))

### getMidiFileFormat

public static [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html) **getMidiFileFormat**([File](http://docs.google.com/java/io/File.html) file)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains the MIDI file format of the specified File. The File must point to valid MIDI file data for a file type recognized by the system.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while determining the file format.

**Parameters:**file - the File from which file format information should be extracted **Returns:**a MidiFileFormat object describing the MIDI file format **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the File does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs while accessing the file**See Also:**[getMidiFileFormat(InputStream)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.io.InputStream)), [getMidiFileFormat(URL)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileFormat(java.net.URL))

### getSequence

public static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) **getSequence**([InputStream](http://docs.google.com/java/io/InputStream.html) stream)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains a MIDI sequence from the specified input stream. The stream must point to valid MIDI file data for a file type recognized by the system.

This method and/or the code it invokes may need to read some data from the stream to determine whether its data format is supported. The implementation may therefore need to mark the stream, read enough data to determine whether it is in a supported format, and reset the stream's read pointer to its original position. If the input stream does not permit this set of operations, this method may fail with an IOException.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while constructing the Sequence object from the file data.

**Parameters:**stream - the input stream from which the Sequence should be constructed **Returns:**a Sequence object based on the MIDI file data contained in the input stream **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the stream does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs while accessing the stream**See Also:**[InputStream.markSupported()](http://docs.google.com/java/io/InputStream.html#markSupported()), [InputStream.mark(int)](http://docs.google.com/java/io/InputStream.html#mark(int))

### getSequence

public static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) **getSequence**([URL](http://docs.google.com/java/net/URL.html) url)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains a MIDI sequence from the specified URL. The URL must point to valid MIDI file data for a file type recognized by the system.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while constructing the Sequence object from the file data.

**Parameters:**url - the URL from which the Sequence should be constructed **Returns:**a Sequence object based on the MIDI file data pointed to by the URL **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the URL does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs while accessing the URL

### getSequence

public static [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) **getSequence**([File](http://docs.google.com/java/io/File.html) file)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Obtains a MIDI sequence from the specified File. The File must point to valid MIDI file data for a file type recognized by the system.

This operation can only succeed for files of a type which can be parsed by an installed file reader. It may fail with an InvalidMidiDataException even for valid files if no compatible file reader is installed. It will also fail with an InvalidMidiDataException if a compatible file reader is installed, but encounters errors while constructing the Sequence object from the file data.

**Parameters:**file - the File from which the Sequence should be constructed **Returns:**a Sequence object based on the MIDI file data pointed to by the File **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if the File does not point to valid MIDI file data recognized by the system [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs

### getMidiFileTypes

public static int[] **getMidiFileTypes**()

Obtains the set of MIDI file types for which file writing support is provided by the system.

**Returns:**array of unique file types. If no file types are supported, an array of length 0 is returned.

### isFileTypeSupported

public static boolean **isFileTypeSupported**(int fileType)

Indicates whether file writing support for the specified MIDI file type is provided by the system.

**Parameters:**fileType - the file type for which write capabilities are queried **Returns:**true if the file type is supported, otherwise false

### getMidiFileTypes

public static int[] **getMidiFileTypes**([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) sequence)

Obtains the set of MIDI file types that the system can write from the sequence specified.

**Parameters:**sequence - the sequence for which MIDI file type support is queried **Returns:**the set of unique supported file types. If no file types are supported, returns an array of length 0.

### isFileTypeSupported

public static boolean **isFileTypeSupported**(int fileType,  
 [Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) sequence)

Indicates whether a MIDI file of the file type specified can be written from the sequence indicated.

**Parameters:**fileType - the file type for which write capabilities are queriedsequence - the sequence for which file writing support is queried **Returns:**true if the file type is supported for this sequence, otherwise false

### write

public static int **write**([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) in,  
 int fileType,  
 [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Writes a stream of bytes representing a file of the MIDI file type indicated to the output stream provided.

**Parameters:**in - sequence containing MIDI data to be written to the filefileType - the file type of the file to be written to the output streamout - stream to which the file data should be written **Returns:**the number of bytes written to the output stream **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the file format is not supported by the system**See Also:**[isFileTypeSupported(int, Sequence)](http://docs.google.com/javax/sound/midi/MidiSystem.html#isFileTypeSupported(int,%20javax.sound.midi.Sequence)), [getMidiFileTypes(Sequence)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileTypes(javax.sound.midi.Sequence))

### write

public static int **write**([Sequence](http://docs.google.com/javax/sound/midi/Sequence.html) in,  
 int type,  
 [File](http://docs.google.com/java/io/File.html) out)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Writes a stream of bytes representing a file of the MIDI file type indicated to the external file provided.

**Parameters:**in - sequence containing MIDI data to be written to the filetype - the file type of the file to be written to the output streamout - external file to which the file data should be written **Returns:**the number of bytes written to the file **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O exception occurs [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the file type is not supported by the system**See Also:**[isFileTypeSupported(int, Sequence)](http://docs.google.com/javax/sound/midi/MidiSystem.html#isFileTypeSupported(int,%20javax.sound.midi.Sequence)), [getMidiFileTypes(Sequence)](http://docs.google.com/javax/sound/midi/MidiSystem.html#getMidiFileTypes(javax.sound.midi.Sequence))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/MidiSystem.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/midi/MidiMessage.html)   [**NEXT CLASS**](http://docs.google.com/javax/sound/midi/MidiUnavailableException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sound/midi/MidiSystem.html)    [**NO FRAMES**](http://docs.google.com/MidiSystem.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | 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).