| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Sequence.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/Receiver.html)   [**NEXT CLASS**](http://docs.google.com/javax/sound/midi/Sequencer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sound/midi/Sequence.html)    [**NO FRAMES**](http://docs.google.com/Sequence.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#44sinio) | [METHOD](#3j2qqm3) |

## **javax.sound.midi**

Class Sequence

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

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

A Sequence is a data structure containing musical information (often an entire song or composition) that can be played back by a [Sequencer](http://docs.google.com/javax/sound/midi/Sequencer.html) object. Specifically, the Sequence contains timing information and one or more tracks. Each [track](http://docs.google.com/javax/sound/midi/Track.html) consists of a series of MIDI events (such as note-ons, note-offs, program changes, and meta-events). The sequence's timing information specifies the type of unit that is used to time-stamp the events in the sequence.

A Sequence can be created from a MIDI file by reading the file into an input stream and invoking one of the getSequence methods of [MidiSystem](http://docs.google.com/javax/sound/midi/MidiSystem.html). A sequence can also be built from scratch by adding new Tracks to an empty Sequence, and adding [MidiEvent](http://docs.google.com/javax/sound/midi/MidiEvent.html) objects to these Tracks.

**See Also:**[Sequencer.setSequence(java.io.InputStream stream)](http://docs.google.com/javax/sound/midi/Sequencer.html#setSequence(java.io.InputStream)), [Sequencer.setSequence(Sequence sequence)](http://docs.google.com/javax/sound/midi/Sequencer.html#setSequence(javax.sound.midi.Sequence)), [Track.add(MidiEvent)](http://docs.google.com/javax/sound/midi/Track.html#add(javax.sound.midi.MidiEvent)), [MidiFileFormat](http://docs.google.com/javax/sound/midi/MidiFileFormat.html)

| **Field Summary** | |
| --- | --- |
| protected  float | [**divisionType**](http://docs.google.com/javax/sound/midi/Sequence.html#divisionType)            The timing division type of the sequence. |
| static float | [**PPQ**](http://docs.google.com/javax/sound/midi/Sequence.html#PPQ)            The tempo-based timing type, for which the resolution is expressed in pulses (ticks) per quarter note. |
| protected  int | [**resolution**](http://docs.google.com/javax/sound/midi/Sequence.html#resolution)            The timing resolution of the sequence. |
| static float | [**SMPTE\_24**](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_24)            The SMPTE-based timing type with 24 frames per second (resolution is expressed in ticks per frame). |
| static float | [**SMPTE\_25**](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_25)            The SMPTE-based timing type with 25 frames per second (resolution is expressed in ticks per frame). |
| static float | [**SMPTE\_30**](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30)            The SMPTE-based timing type with 30 frames per second (resolution is expressed in ticks per frame). |
| static float | [**SMPTE\_30DROP**](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30DROP)            The SMPTE-based timing type with 29.97 frames per second (resolution is expressed in ticks per frame). |
| protected  [Vector](http://docs.google.com/java/util/Vector.html)<[Track](http://docs.google.com/javax/sound/midi/Track.html)> | [**tracks**](http://docs.google.com/javax/sound/midi/Sequence.html#tracks)            The MIDI tracks in this sequence. |

| **Constructor Summary** | |
| --- | --- |
| [**Sequence**](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int))(float divisionType, int resolution)            Constructs a new MIDI sequence with the specified timing division type and timing resolution. |
| [**Sequence**](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int,%20int))(float divisionType, int resolution, int numTracks)            Constructs a new MIDI sequence with the specified timing division type, timing resolution, and number of tracks. |

| **Method Summary** | |
| --- | --- |
| [Track](http://docs.google.com/javax/sound/midi/Track.html) | [**createTrack**](http://docs.google.com/javax/sound/midi/Sequence.html#createTrack())()            Creates a new, initially empty track as part of this sequence. |
| boolean | [**deleteTrack**](http://docs.google.com/javax/sound/midi/Sequence.html#deleteTrack(javax.sound.midi.Track))([Track](http://docs.google.com/javax/sound/midi/Track.html) track)            Removes the specified track from the sequence. |
| float | [**getDivisionType**](http://docs.google.com/javax/sound/midi/Sequence.html#getDivisionType())()            Obtains the timing division type for this sequence. |
| long | [**getMicrosecondLength**](http://docs.google.com/javax/sound/midi/Sequence.html#getMicrosecondLength())()            Obtains the duration of this sequence, expressed in microseconds. |
| [Patch](http://docs.google.com/javax/sound/midi/Patch.html)[] | [**getPatchList**](http://docs.google.com/javax/sound/midi/Sequence.html#getPatchList())()            Obtains a list of patches referenced in this sequence. |
| int | [**getResolution**](http://docs.google.com/javax/sound/midi/Sequence.html#getResolution())()            Obtains the timing resolution for this sequence. |
| long | [**getTickLength**](http://docs.google.com/javax/sound/midi/Sequence.html#getTickLength())()            Obtains the duration of this sequence, expressed in MIDI ticks. |
| [Track](http://docs.google.com/javax/sound/midi/Track.html)[] | [**getTracks**](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks())()            Obtains an array containing all the tracks in this sequence. |

| **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** |
| --- |

### PPQ

public static final float **PPQ**

The tempo-based timing type, for which the resolution is expressed in pulses (ticks) per quarter note.

**See Also:**[Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [Constant Field Values](http://docs.google.com/constant-values.html#javax.sound.midi.Sequence.PPQ)

### SMPTE\_24

public static final float **SMPTE\_24**

The SMPTE-based timing type with 24 frames per second (resolution is expressed in ticks per frame).

**See Also:**[Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [Constant Field Values](http://docs.google.com/constant-values.html#javax.sound.midi.Sequence.SMPTE_24)

### SMPTE\_25

public static final float **SMPTE\_25**

The SMPTE-based timing type with 25 frames per second (resolution is expressed in ticks per frame).

**See Also:**[Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [Constant Field Values](http://docs.google.com/constant-values.html#javax.sound.midi.Sequence.SMPTE_25)

### SMPTE\_30DROP

public static final float **SMPTE\_30DROP**

The SMPTE-based timing type with 29.97 frames per second (resolution is expressed in ticks per frame).

**See Also:**[Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [Constant Field Values](http://docs.google.com/constant-values.html#javax.sound.midi.Sequence.SMPTE_30DROP)

### SMPTE\_30

public static final float **SMPTE\_30**

The SMPTE-based timing type with 30 frames per second (resolution is expressed in ticks per frame).

**See Also:**[Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [Constant Field Values](http://docs.google.com/constant-values.html#javax.sound.midi.Sequence.SMPTE_30)

### divisionType

protected float **divisionType**

The timing division type of the sequence.

**See Also:**[PPQ](http://docs.google.com/javax/sound/midi/Sequence.html#PPQ), [SMPTE\_24](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_24), [SMPTE\_25](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_25), [SMPTE\_30DROP](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30DROP), [SMPTE\_30](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30), [getDivisionType()](http://docs.google.com/javax/sound/midi/Sequence.html#getDivisionType())

### resolution

protected int **resolution**

The timing resolution of the sequence.

**See Also:**[getResolution()](http://docs.google.com/javax/sound/midi/Sequence.html#getResolution())

### tracks

protected [Vector](http://docs.google.com/java/util/Vector.html)<[Track](http://docs.google.com/javax/sound/midi/Track.html)> **tracks**

The MIDI tracks in this sequence.

**See Also:**[getTracks()](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks())

| **Constructor Detail** |
| --- |

### Sequence

public **Sequence**(float divisionType,  
 int resolution)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html)

Constructs a new MIDI sequence with the specified timing division type and timing resolution. The division type must be one of the recognized MIDI timing types. For tempo-based timing, divisionType is PPQ (pulses per quarter note) and the resolution is specified in ticks per beat. For SMTPE timing, divisionType specifies the number of frames per second and the resolution is specified in ticks per frame. The sequence will contain no initial tracks. Tracks may be added to or removed from the sequence using [createTrack()](http://docs.google.com/javax/sound/midi/Sequence.html#createTrack()) and [deleteTrack(javax.sound.midi.Track)](http://docs.google.com/javax/sound/midi/Sequence.html#deleteTrack(javax.sound.midi.Track)).

**Parameters:**divisionType - the timing division type (PPQ or one of the SMPTE types)resolution - the timing resolution **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if divisionType is not valid**See Also:**[PPQ](http://docs.google.com/javax/sound/midi/Sequence.html#PPQ), [SMPTE\_24](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_24), [SMPTE\_25](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_25), [SMPTE\_30DROP](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30DROP), [SMPTE\_30](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30), [getDivisionType()](http://docs.google.com/javax/sound/midi/Sequence.html#getDivisionType()), [getResolution()](http://docs.google.com/javax/sound/midi/Sequence.html#getResolution()), [getTracks()](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks())

### Sequence

public **Sequence**(float divisionType,  
 int resolution,  
 int numTracks)  
 throws [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html)

Constructs a new MIDI sequence with the specified timing division type, timing resolution, and number of tracks. The division type must be one of the recognized MIDI timing types. For tempo-based timing, divisionType is PPQ (pulses per quarter note) and the resolution is specified in ticks per beat. For SMTPE timing, divisionType specifies the number of frames per second and the resolution is specified in ticks per frame. The sequence will be initialized with the number of tracks specified by numTracks. These tracks are initially empty (i.e. they contain only the meta-event End of Track). The tracks may be retrieved for editing using the [getTracks()](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks()) method. Additional tracks may be added, or existing tracks removed, using [createTrack()](http://docs.google.com/javax/sound/midi/Sequence.html#createTrack()) and [deleteTrack(javax.sound.midi.Track)](http://docs.google.com/javax/sound/midi/Sequence.html#deleteTrack(javax.sound.midi.Track)).

**Parameters:**divisionType - the timing division type (PPQ or one of the SMPTE types)resolution - the timing resolutionnumTracks - the initial number of tracks in the sequence. **Throws:** [InvalidMidiDataException](http://docs.google.com/javax/sound/midi/InvalidMidiDataException.html) - if divisionType is not valid**See Also:**[PPQ](http://docs.google.com/javax/sound/midi/Sequence.html#PPQ), [SMPTE\_24](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_24), [SMPTE\_25](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_25), [SMPTE\_30DROP](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30DROP), [SMPTE\_30](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30), [getDivisionType()](http://docs.google.com/javax/sound/midi/Sequence.html#getDivisionType()), [getResolution()](http://docs.google.com/javax/sound/midi/Sequence.html#getResolution())

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

### getDivisionType

public float **getDivisionType**()

Obtains the timing division type for this sequence.

**Returns:**the division type (PPQ or one of the SMPTE types)**See Also:**[PPQ](http://docs.google.com/javax/sound/midi/Sequence.html#PPQ), [SMPTE\_24](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_24), [SMPTE\_25](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_25), [SMPTE\_30DROP](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30DROP), [SMPTE\_30](http://docs.google.com/javax/sound/midi/Sequence.html#SMPTE_30), [Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [MidiFileFormat.getDivisionType()](http://docs.google.com/javax/sound/midi/MidiFileFormat.html#getDivisionType())

### getResolution

public int **getResolution**()

Obtains the timing resolution for this sequence. If the sequence's division type is PPQ, the resolution is specified in ticks per beat. For SMTPE timing, the resolution is specified in ticks per frame.

**Returns:**the number of ticks per beat (PPQ) or per frame (SMPTE)**See Also:**[getDivisionType()](http://docs.google.com/javax/sound/midi/Sequence.html#getDivisionType()), [Sequence(float, int)](http://docs.google.com/javax/sound/midi/Sequence.html#Sequence(float,%20int)), [MidiFileFormat.getResolution()](http://docs.google.com/javax/sound/midi/MidiFileFormat.html#getResolution())

### createTrack

public [Track](http://docs.google.com/javax/sound/midi/Track.html) **createTrack**()

Creates a new, initially empty track as part of this sequence. The track initially contains the meta-event End of Track. The newly created track is returned. All tracks in the sequence may be retrieved using [getTracks()](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks()). Tracks may be removed from the sequence using [deleteTrack(javax.sound.midi.Track)](http://docs.google.com/javax/sound/midi/Sequence.html#deleteTrack(javax.sound.midi.Track)).

**Returns:**the newly created track

### deleteTrack

public boolean **deleteTrack**([Track](http://docs.google.com/javax/sound/midi/Track.html) track)

Removes the specified track from the sequence.

**Parameters:**track - the track to remove **Returns:**true if the track existed in the track and was removed, otherwise false.**See Also:**[createTrack()](http://docs.google.com/javax/sound/midi/Sequence.html#createTrack()), [getTracks()](http://docs.google.com/javax/sound/midi/Sequence.html#getTracks())

### getTracks

public [Track](http://docs.google.com/javax/sound/midi/Track.html)[] **getTracks**()

Obtains an array containing all the tracks in this sequence. If the sequence contains no tracks, an array of length 0 is returned.

**Returns:**the array of tracks**See Also:**[createTrack()](http://docs.google.com/javax/sound/midi/Sequence.html#createTrack()), [deleteTrack(javax.sound.midi.Track)](http://docs.google.com/javax/sound/midi/Sequence.html#deleteTrack(javax.sound.midi.Track))

### getMicrosecondLength

public long **getMicrosecondLength**()

Obtains the duration of this sequence, expressed in microseconds.

**Returns:**this sequence's duration in microseconds.

### getTickLength

public long **getTickLength**()

Obtains the duration of this sequence, expressed in MIDI ticks.

**Returns:**this sequence's length in ticks**See Also:**[getMicrosecondLength()](http://docs.google.com/javax/sound/midi/Sequence.html#getMicrosecondLength())

### getPatchList

public [Patch](http://docs.google.com/javax/sound/midi/Patch.html)[] **getPatchList**()

Obtains a list of patches referenced in this sequence. This patch list may be used to load the required [Instrument](http://docs.google.com/javax/sound/midi/Instrument.html) objects into a [Synthesizer](http://docs.google.com/javax/sound/midi/Synthesizer.html).

**Returns:**an array of [Patch](http://docs.google.com/javax/sound/midi/Patch.html) objects used in this sequence**See Also:**[Synthesizer.loadInstruments(Soundbank, Patch[])](http://docs.google.com/javax/sound/midi/Synthesizer.html#loadInstruments(javax.sound.midi.Soundbank,%20javax.sound.midi.Patch%5B%5D))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Sequence.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/Receiver.html)   [**NEXT CLASS**](http://docs.google.com/javax/sound/midi/Sequencer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sound/midi/Sequence.html)    [**NO FRAMES**](http://docs.google.com/Sequence.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#44sinio) | [METHOD](#3j2qqm3) |

[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).