MiscSoundTracker

Inherits From: Object

Declared In: MiscSoundTracker.h

Class Description

The MiscSoundTracker tracks a variety of real-time sound-playing or sound-recording information. In real-time, it can keep an application up-to-date as to:

- How many samples a sound has played or recorded so far.
- How many seconds a sound has played or recorded so far.
- What percent of a sound has been played so far, expressed as a number from 0 to 100.
- The fraction (ratio) of a sound played so far, expressed as a number from 0 to 1.
- The current volume of the left output channel.

- The current volume of the right output channel.
- The current volume of both output channels combined.
- The current mute status.
- The current deemphasis status.
- The current peak volume of the left output channel.
- The current peak volume of the right output channel.
- The current peak volume of both output channels combined.

One MiscSoundTracker can track one and only one of the above at a time, but you're free to create as many MiscSoundTrackers as necessary to get the job done.

You create a MiscSoundTracker with the **init** method. Next, you need to provide the MiscSoundTracker with an target to update, using **setTarget:.** This target must be able to respond to **takeFloatValueFrom:** and **takeIntValueFrom:,** and either **setStringValue:** or **setIntValue:**, or both, just like a subclass of Control or Cell. It should respond to the **take...From:** methods by calling the MiscSoundTracker's **floatValue** or **intValue** methods to be updated.

Now, indicate what kind of tracking you'd like your MiscSoundTracker to perform, using one of a variety of **send...** methods. If you're tracking the samples, seconds, or percent played or recorded from a particular sound (you chose **sendSamples, sendSeconds, sendPercent,** or **sendRatio**), you'll also need to indicate the sound to track with the **setSound:** method.

Now start your MiscSoundTracker with **run**. At any time you can stop or suspend the MiscSoundTracker with **stop.** After suspending it, you're free to call **run** again to resume tracking. Every so often (the default is

SOUNDTRACKER_TIMED_ENTRY_SPEED, about 2 seconds) the MiscSoundTracker will call its target to update it with new information. If you've chosen **sendSamples,** the MiscSoundTracker will relay the number of samples to your target using the standard **takeIntValueFrom:** method. If you've chosen **sendMute** or **sendDeemphasis,** the MiscSoundTracker relays YES or NO (as 1 or 0) using **takeIntValueFrom:**. For all other tracking formats, the MiscSoundTracker relays a number to the target using **takeFloatValueFrom:**.

If you're tracking volume, mute, or deemphasis, you can use the MiscSoundTracker to set these values as well as track them, using **setVolumeTo:, setMuteTo:,** or **setDeemphasisTo:.**

When the MiscSoundTracker has *no* information to give the target (for example, when a sound isn't playing or recording and you're tracking samples), it will attempt to *clear* the target. If your target responds to the **setStringValue:** method implemented by Control and Cell, the MiscSoundTracker will clear it by setting its string value to "". If the target does not respond to **setStringValue,** the MiscTracker will clear it by using **setIntValue:** to set the target to 0.

Some important notes: A bug in NeXT's Sound object prevents us from finding out what the sampling rate of a sound being recorded is. As a result, if you've chosen **sendSeconds,** MiscSoundTracker must rely on an assumed "default" sampling rate to compute the number of seconds recorded so far. The default "default" rate is SOUNDTRACKER_DEFAULT_SAMPLING_RATE (44100), which is proper for recording from the NeXT microphone. If for some reason you're recording with some other format but still through NeXT's NXSoundStream system, you can change this rate with **setDefaultSamplingRate:.** All this is unneccessary when playing the sound.

Although you can track the samples and seconds of a sound being recorded, if you think about it, you'll see that

you can't track the *percent* of a sound recorded so far. Thus, **sendRatio** and **sendPercent** only work for sounds being *played.*

Bugs: NeXT's sound objects do not accurately say when they've finished playing a sound. As a result, if the MiscSoundTracker is tracking samples, seconds, or percent played, it may stop tracking a full second before the sound actually stops playing! For small sounds the MiscSoundTracker may not track at all. If anyone knows how to get NeXTSTEP to provide more accurate information, please give me a ring at seanl@cs.umd.edu.

Version Incompatibility: Versions later than 1.1 of the MiscSoundTracker's archiving format are incompatible with earlier versions.

Instance Variables

id	**sound;**
id	**target;**
BOOL	**running;**
int	**send_type;**
float	**refresh;**
DPSTimedEntry	**teNum;**
NXSoundOut*	**output_device;**
float	**default_sampling_rate;**

sound The sound, if any, the MiscSoundTracker is tracking

target	The MiscSoundTracker's target to update
running	Is the MiscSoundTracker currently running?
send_type	The type of tracking the MiscSoundTracker is doing
refresh	The number of seconds between updates
teNum	The MiscSoundTracker's timed entry flag
output_device	The MiscSoundTracker's private output device for sound info queries
default_sampling_rate	The rate to use when recording a sound and tracking seconds

Method Types

Creating and freeing instances	- init - free
Setting Parameters	- setRefresh: - refresh; - setDefaultSamplingRate: - defaultSamplingRate;
Setting a Sound	- setSound:

	- sound
Setting or informing the target	- setTarget: - target - clearTarget
Starting and stopping the tracker	- run - run: - stop - stop:
Setting the tracking type	- sendSamples - sendSamples: - sendSeconds - sendSeconds: - sendRatio - sendRatio: - sendPercent - sendPercent: - sendLeft - sendLeft: - sendRight - sendRight: - sendMono

	- sendMono: - sendMute - sendMute: - sendDeemphasis - sendDeemphasis: - sendLeftPeak - sendLeftPeak: - sendRightPeak - sendRightPeak: - sendMonoPeak - sendMonoPeak: - setSendType:
Querying the tracking type	- sendType
Modifying common parameters	- setVolumeTo: - setMuteTo: - setDeemphasisTo:
Responding to target inquiries	- floatValue - intValue
Archiving	- read: - write:

Instance Methods

clearTarget
- **clearTarget**

Clears the target's value by either setting its string value to "" (if the target responds to **setStringValue:**), or setting its integer value to 0.

See also: -€**target**, -€**setTarget:**

defaultSamplingRate
- (float)**defaultSamplingRate**

Returns the MiscSoundTracker's default sampling rate. This default sampling rate is used exclusively to track seconds information in recorded sounds.

See also: -€**setDefaultSamplingRate:**

floatValue
- (float) **floatValue**

Returns the current MiscSoundTracker update value in floating-point.

See also: -€**intValue**

free
- **free**

Stops tracking, and frees the MiscSoundTracker its private sound device.

See also: -€**free** (Object)

init
- **init**

Initializes the MiscSoundTracker and generates its private sound device.

See also: -€**init** (Object)

intValue
- (int) **intValue**

Returns the current MiscSoundTracker update value as an integer.

See also: -€**floatValue**

read:

- **read:**(NXTypedStream*) *stream*

Reads in the MiscSoundTracker from archived state in *stream*.

See also: -€**write:**

refresh

- (float)**refresh**

Returns the refresh rate. The refresh rate is the number of seconds between calls to the target to update its sound information. The default is SOUNDTRACKER_TIMED_ENTRY_SPEED, about 2 seconds.

See also: **± setRefresh:**

run

- **run**

Runs or resumes the MiscSoundTracker.

See also: -€**run:**, -€**stop**, -€**stop:**

run:

- **run:***sender*

Runs or resumes the MiscSoundTracker.

See also: -€**run**, -€**stop**, -€**stop:**

sendDeemphasis

- **sendDeemphasis**

Tells the MiscSoundTracker deemphasis status, expressed as a YES or NO value (0 or 1). This information is relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendDeemphasis:

- **sendDeemphasis:***sender*

Tells the MiscSoundTracker deemphasis status, expressed as a YES or NO value (0 or 1). This information is

relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendLeft

- **sendLeft**

Tells the MiscSoundTracker to track the amplitude of the left output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendLeft:

- **sendLeft:***sender*

Tells the MiscSoundTracker to track the amplitude of the left output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendLeftPeak

- **sendLeftPeak**

Tells the MiscSoundTracker to track the peak amplitude of the left output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendLeftPeak:

- **sendLeftPeak:***sender*

Tells the MiscSoundTracker to track the peak amplitude of the left output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMono

- **sendMono**

Tells the MiscSoundTracker to track the amplitudes of both the left and right output channels averaged together, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMono:

- **sendMono:***sender*

Tells the MiscSoundTracker to track the amplitudes of both the left and right output channels averaged together, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMonoPeak

- **sendMonoPeak**

Tells the MiscSoundTracker to track the peak amplitudes of both the left and right output channels averaged together, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMonoPeak:

- **sendMonoPeak:***sender*

Tells the MiscSoundTracker to track the peak amplitudes of both the left and right output channels averaged together, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMute
- **sendMute**

Tells the MiscSoundTracker mute status, expressed as a YES or NO value (0 or 1). This information is relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendMute:
- **sendMute:***sender*

Tells the MiscSoundTracker mute status, expressed as a YES or NO value (0 or 1). This information is relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendPercent
- **sendPercent**

Tells the MiscSoundTracker to track the percent played from a sound, from 0 to 100. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendPercent:

- **sendPercent:***sender*

Tells the MiscSoundTracker to track the percent played from a sound, from 0 to 100. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRatio

- **sendRatio**

Tells the MiscSoundTracker to track the fraction played from a sound, from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRatio:

- **sendRatio:***sender*

Tells the MiscSoundTracker to track the fraction played from a sound, from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRight

- **sendRight**

Tells the MiscSoundTracker to track the amplitude of the right output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRight:

- **sendRight:***sender*

Tells the MiscSoundTracker to track the amplitude of the right output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRightPeak

- **sendRightPeak**

Tells the MiscSoundTracker to track the peak amplitude of the right output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendRightPeak:

- **sendRightPeak:***sender*

Tells the MiscSoundTracker to track the peakamplitude of the right output channel, expressed as a number from 0 to 1. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendSamples

- **sendSamples**

Tells the MiscSoundTracker to track samples played or recorded from a sound. This information is relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendSamples:
- **sendSamples:***sender*

Tells the MiscSoundTracker to track samples played or recorded from a sound. This information is relayed to the target using **takeIntValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendSeconds
- **sendSeconds**

Tells the MiscSoundTracker to track seconds played or recorded from a sound. This information is relayed to the target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendSeconds:
- **sendSeconds:***sender*

Tells the MiscSoundTracker to track seconds played or recorded from a sound. This information is relayed to the

target using **takeFloatValueFrom**.

See also all other methods of the form -€send... or -€send...:

sendType

- (int) **sendType**

Returns the MiscSoundTracker's sending/tracking type. Formats include:

SOUNDTRACKER_SEND_TYPE_SAMPLES
SOUNDTRACKER_SEND_TYPE_SECONDS
SOUNDTRACKER_SEND_TYPE_RATIO
SOUNDTRACKER_SEND_TYPE_PERCENT
SOUNDTRACKER_SEND_TYPE_LEFT
SOUNDTRACKER_SEND_TYPE_RIGHT
SOUNDTRACKER_SEND_TYPE_MONO
SOUNDTRACKER_SEND_TYPE_MUTE
SOUNDTRACKER_SEND_TYPE_DEEMPHASIS
SOUNDTRACKER_SEND_TYPE_LEFT_PEAK
SOUNDTRACKER_SEND_TYPE_RIGHT_PEAK
SOUNDTRACKER_SEND_TYPE_MONO_PEAK

See also: -€setSendType:

setDefaultSamplingRate:

- **setDefaultSamplingRate:**(float) *this_rate*

Sets the MiscSoundTracker's default sampling rate to *this_rate.* This default sampling rate is used exclusively to track seconds information in recorded sounds. If for some reason the your software does not appear to track seconds properly when recorded, you can change this value from its default value of SOUNDTRACKER_DEFAULT_SAMPLING_RATE (44100, which should work in most cases, including the NeXT microphone). In reality, you'll probably never change it. *this_rate* should never be 0.

See also: -€**defaultSamplingRate**

setRefresh:

- **setRefresh:**(float) *number_seconds*

Sets the refresh rate to *number_seconds.* The refresh rate is the number of seconds between calls to the target to update its sound information. The default is SOUNDTRACKER_TIMED_ENTRY_SPEED, about 2 seconds.

See also: ± refresh

setSendType:

- **setSendType:**(int)*this_format*

Tells the MiscSoundTracker to track *this_format.* Formats include:

SOUNDTRACKER_SEND_TYPE_SAMPLES
SOUNDTRACKER_SEND_TYPE_SECONDS
SOUNDTRACKER_SEND_TYPE_RATIO
SOUNDTRACKER_SEND_TYPE_PERCENT
SOUNDTRACKER_SEND_TYPE_LEFT
SOUNDTRACKER_SEND_TYPE_RIGHT
SOUNDTRACKER_SEND_TYPE_MONO
SOUNDTRACKER_SEND_TYPE_MUTE
SOUNDTRACKER_SEND_TYPE_DEEMPHASIS
SOUNDTRACKER_SEND_TYPE_LEFT_PEAK
SOUNDTRACKER_SEND_TYPE_RIGHT_PEAK
SOUNDTRACKER_SEND_TYPE_MONO_PEAK

If *this_format* is not one of the above formats, **sendType:** fails and returns NULL.

See also all other methods of the form -€send... or -€send...:, ± sendType

setSound:

- **setSound:***this_sound_or_soundview*

Sets the sound to track to *this_sound_or_soundview,* which may be a Sound or a SoundView. If it's a SoundView, the MiscSoundTracker queries the SoundView for its sound and tracks that. If

this_sound_or_soundview is NULL, the MiscSoundTracker is freed from tracking a sound.

See also: -€**sound**

setTarget:
- **setTarget:***this_target*

Sets the target to *this_target.* *this_target* must be able to respond to **takeFloatValueFrom:** and **takeIntValueFrom:,** and either **setStringValue:** or **setIntValue:**, or both, just like a subclass of Control or Cell. It should respond to the **take...From:** methods by calling the MiscSoundTracker's **floatValue** or **intValue** methods to be updated. If *this_target* is NULL, the MiscSoundTracker is set to no target.

See also: -€**target, -€clearTarget, -€floatValue, -€intValue**

setDeemphasisTo:
- **setDeemphasisTo:***sender*

Tells the MiscSoundTracker to query the sender (using **intValue**) for a YES or NO value (1 or 0). The MiscSoundTracker then sets the NeXT's deemphasis to this value.

See also: -€**setVolumeTo:, -€setMuteTo:**

setMuteTo:

- **setMuteTo:***sender*

Tells the MiscSoundTracker to query the sender (using **intValue**) for a YES or NO value (1 or 0). The MiscSoundTracker then sets the NeXT's mute to this value.

See also: -€**setVolumeTo:**, -€**setDeemphasisTo:**

setVolumeTo:

- **setVolumeTo:***sender*

Tells the MiscSoundTracker to query the sender (using **floatValue**) for a number. The MiscSoundTracker then sets the NeXT's output volume to this value.

See also: -€**setMuteTo:**, -€**setDeemphasisTo:**

sound

- **sound**

Returns the current sound to that the MiscSoundTracker is tracking. If the MiscSoundTracker is tracking no sound, returns NULL.

See also: -€**setSound:**

stop
- **stop**

Stops or suspends the MiscSoundTracker.

See also: -€**run:**, -€**run**, -€**stop:**

stop:
- **stop:***sender*

Stops or suspends the MiscSoundTracker.

See also: -€**run**, -€**stop**, -€**run:**

target
- **target**

Returns the MiscSoundTracker's target. If the MiscSoundTracker has no target, returns NULL.

See also: -€**setTarget:**, -€**clearTarget**

write:

- **write:**(NXTypedStream*) *stream*

Archives the MiscSoundTracker to *stream*.

See also: -€**read:**