MiscVolumeMeter

Inherits From: View

Declared In: MiscVolumeMeter.h

Class Description

The MiscVolumeMeter is a replacement for NeXT's SoundMeter object. The MiscVolumeMeter allows for both vertical and horizontal bar positions, stereo display, and the ability to either meter one sound or all sounds.

Create a MiscVolumeMeter using **initFrame:**, optionally lock it onto a sound with **setSound:**, set whatever auxillary features are necessary, and start it running with **run**. You can pause the meter with **stop** and resume it again with **run** if you like. If you lock the MiscVolumeMeter onto a sound, it will adjust itself to reflect the qualities of the sound (recording or playing, stereo or mono). If you do not lock the MiscVolumeMeter onto a sound, you need to set these yourself. If a MiscVolumeMeter is not locked onto a sound, it will reflect playing whenever any sound from any application is playing out through the speaker, or (if it's reflecting input instead) reflect recording whenver any sound is recording. It's sometimes convenient to have the MiscVolumeMeter *constantly* reflect any input received by the microphone, regardless of whether or not a sound is recording. To

do this, use the MiscVolumeMeter in conjunction with the MiscTapper object.

The MiscVolumeMeter will display horizontally (0 volume at left, max volume at right, left channel on top) if it is wider than tall. It will display vertically if it is taller than wide. If the MiscVolumeMeter feels it doesn't have enough room to draw anything, it will not draw.

The MiscVolumeMeter can be set to turn itself off automatically whenever the window it is in is hidden or closed, or on whenever the window is displayed. This conserves valuable resources, as the MiscVolumeMeter is fairly heavy on system load when running. To do this, set the MiscVolumeMeter in question as the delegate of its window.

The MiscVolumeMeter can send methods to a delegate, indicating that the MiscVolumeMeter redrew itself. This is useful to tie some other checking of a sound into the MiscVolumeMeter's timed-entry function.

Bugs: Because of NeXT's sound-mixing strategy in versions 3.x and beyond, the MiscVolumeMeter cannot distinguish between the output of different sounds, even sounds in different applications. If you locked the MiscVolumeMeter on a sound, started playing the sound, and some other application played *another* sound at the same time, the MiscVolumeMeter would *also* reflect that sound's output combined with the output of the locked sound. This can't be avoided without twiddling with the internal workings of the NeXT sound object, which I'm not about to do. :-)

Also, NeXT's sound objects do not accurately say when they've finished playing a sound. As a result, if the MiscVolumeMeter has been locked onto a playing sound, the meter may stop drawing up to a full second before the sound stops playing! For small sounds the MiscVolumeMeter may not draw at all. NeXT's SoundMeter apparently uses another private method of checking up on sounds, so to my knowledge there's no way around this. If anyone knows, please give me a ring at seanl@cs.umd.edu.

Version Incompatibility: Versions later than 1.2 of the MiscVolumeMeter's archiving format are incompatible with earlier versions.

Instance Variables

NXSoundIn*	**input_device;**
NXSoundOut*	**output_device;**
id	**sound;**
id	**delegate;**
BOOL	**input;**
BOOL	**running;**
BOOL	**bezeled;**
BOOL	**peak_bubble_displayed;**
BOOL	**stereo;**
float	**background_gray;**
float	**value_gray;**
float	**bubble_gray;**
float	**refresh;**
int	**refreshes_per_new_peak_bubble;**
int	**refresh_tally;**
float	**refreshes_left[];**
float	**refreshes_right[];**
int	**current_max_refresh_left;**
int	**current_max_refresh_right;**
DPSTimedEntry	**teNum;**

input_device	The NXSoundIn device to take input information from.
output_device	The NXSoundOut device to take output information from.
sound	The sound, if any, the MiscVolumeMeter is locked onto.
delegate	The MiscVolumeMeter's delegate
input	Is the MiscVolumeMeter relfecting input/recording (as opposed to ouput/playing)?
running	Is the MiscVolumeMeter running?
bezeled	Is the MiscVolumeMeter bezeled?
peak_bubble_displayed	Is the MiscVolumeMeter's peak bubble displayed?
stereo	Is the MiscVolumeMeter in stereo?
background_gray	The background gray of the MiscVolumeMeter.
value_gray	The gray of the MiscVolumeMeter's bar.
bubble_gray	The gray of the MiscVolumeMeter's peak bubble.
refresh	The time in-between updates to MiscVolumeMeter's display.
refreshes_per_new_peak_bubble	How many updates to the MiscVolumeMeter's display before updating the bubble.
refresh_tally	Current number of refreshes prior to a bubble update.

refreshes_left	A refresh history to grab the maximum refresh from.
refreshes_right	A refresh history to grab the maximum refresh from.
current_max_refresh_left	Current maximum left value.
current_max_refresh_right	Current maximum right value.
teNum	Timed entry tag.

Method Types

Creating and freeing instances	±€initFrame: ±€free
Displaying the meter	± drawSelf: rects:
Setting parameters ± setMono	± setDelegate: ± setMono: ± setStereo ± setStereo: ± setBackgroundGray: ± setValueGray: ± setBubbleGray: ± setBezeled: ± setPeakBubbleDisplayed: ± setToInput

	± setToInput: ± setToOutput ± setToOutput: ± setRefresh: ± setRefreshesPerNewPeakBubble:
Querying the object	± delegate: ± isStereo: ± backgroundGray: ± valueGray: ± bubbleGray: ± isBezeled: ± peakBubbleDisplayed: ± isInput: ± refresh: ± refreshesPerPeakBubble:
Locking on a Sound	± setSound: ± sound
Operating the meter	± run ± run: ± stop ± stop: ± reclaim
Acting as a Window delegate	± windowDidBecomeKey:

	± windowDidBecomeMain: ± windowDidDeminiaturize: ± windowDidMiniaturize: ± windowWillClose:
Archiving	± read: ± write:

Instance Methods

backgroundGray
- (float) **backgroundGray**

Returns the background gray of the MiscVolumeMeter.

See also: -€**setValueGray:**, -€**setBubbleGray:**, -€**setBackgroundGray**, -€**valueGray**, -€**bubbleGray**

bubbleGray
- (float) **bubbleGray**

Returns the gray of the MiscVolumeMeter's peak bubble.

See also: -€**setValueGray:**, -€**setBubbleGray:**, -€**setBackgroundGray**, -€**valueGray**, -€**backgroundGray**

delegate

- **delegate**

Returns the MiscVolumeMeter's delegate, NULL if none.

See also: -€**setDelegate:**

drawSelf:

- **drawSelf:**(const NXRect*)*rects* :(int)*rectCount*

Draws the MiscVolumeMeter based on current sound information.

See also: -€**drawSelf:rects:** (View)

free

- **free**

Stops the MiscVolumeMeter, frees its devices, and frees the MiscVolumeMeter.

See also: -€**free** (View)

initFrame:

- **initFrame:**(const NXRect*)*frameRect*

Initialized the MiscVolumeMeter and allocates its input and output devices. If the frame is taller than wide, the MiscVolumeMeter displays vertically, else horizontally.

See also: -€**initFrame:** (View)

isBezeled:

- (BOOL) **isBezeled:***sender*

Returns TRUE if displaying with a bezel, FALSE otherwise.

See also: -€**setBezeled**

isInput:

- (BOOL) **isInput:***sender*

Returns TRUE if listening to input/recording, FALSE if listening to output/playing.

See also: -€**setToInput, ± setToOutput, ± setToInput:, ± setToOutput**

isStereo:

- (BOOL) **isStereo:***sender*

Returns TRUE if displaying in stereo, FALSE otherwise.

See also: -€**setMono, ± setStereo**

peakBubbleDisplayed:

- (BOOL) **peakBubbleDisplayed:***sender*

Returns TRUE if displaying the peak bubble, FALSE otherwise.

See also: -€**setPeakBubbleDisplayed**

read:

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

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

See also: -€**write:**

reclaim

- **reclaim**

Attempts to regain control over sound devices. This is rarely (if ever) needed internally, and you shouldn't ever need to call this.

See also:

refresh:

- (float) **refresh:***sender*

Returns the number of seconds (roughly) between updates to the MiscVolumeMeter's information.

See also: -€**setRefresh:, ± setRefreshesPerNewPeakBubble, ±refreshesPerNewPeakBubble:**

refreshesPerPeakBubble

- (int) **refreshesPerPeakBubble:***sender*

Sets the number of refreshes (updates) before a new peak bubble position is calculated.

See also: -€**setRefresh:, ± setRefreshesPerNewPeakBubble, ±refresh**

run

- **run**

Runs or resumes the MiscVolumeMeter.

See also: -€**run:, ± stop, ± stop:**

run:

- **run:***sender*

Runs or resumes the MiscVolumeMeter.

See also: -€**run, ± stop, ± stop:**

setBackgroundGray:

- **setBackgroundGray:**(float)*this_value*

Sets the MiscVolumeMeter's background gray to *this_value,* which should be between 0 and 1, inclusive.

See also: -€**setValueGray:**, -€**setBubbleGray:**, -€**backgroundGray**, -€**valueGray**, -€**bubbleGray**

setBezeled:

- **setBezeled:**(BOOL) *yes_or_no*

Tells the MiscVolumeMeter whether or not to display itself bezeled.

See also: -€**isbezeled:**

setBubbleGray:

- **setBubbleGray:**(float)*this_value*

Sets the gray of the MiscVolumeMeter's peak bubble to *this_value,* which should be between 0 and 1, inclusive.

See also: -€**setBackgroundGray:**, -€**setValueGray:**, -€**backgroundGray**, -€**valueGray**, -€**bubbleGray**

setDelegate:

- **setDelegate:***this_delegate*

Sets the MiscVolumeMeter's delegate. If *this_delegate* is NULL, the MiscVolumeMeter is set to no delegate at all.

See also: -€**delegate**

setMono
- **setMono**

Sets the MiscVolumeMeter to display in mono. If locked onto a sound, the MiscVolumeMeter should ignore this.

See also: -€**mono**, -€**setStereo**, -€**stereo**, -€**setMono:**, -€**setStereo:**

setMono:
- **setMono:***sender*

Sets the MiscVolumeMeter to display in mono. If locked onto a sound, the MiscVolumeMeter should ignore this.

See also: -€**mono**, -€**setStereo**, -€**stereo**, -€**setMono**, -€**setStereo:**

setPeakBubbleDisplayed:
- **setPeakBubbleDisplayed:**(BOOL) *yes_or_no*

Tells the MiscVolumeMeter whether or not to display its peak bubble.

See also: -€**peakBubbleDisplayed:**

setRefresh:

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

Sets the number of seconds (roughly) between updates to the MiscVolumeMeter's information. The default is VOLUMEMETER_TIMED_ENTRY_SPEED, equal to 0.1.

See also: -€**refresh:, ± setRefreshesPerNewPeakBubble, ±refreshesPerNewPeakBubble:**

setRefreshesPerNewPeakBubble:
- **setRefreshesPerNewPeakBubble:**(int) *number_refreshes*

Sets the number of refreshes (updates) before a new peak bubble position is calculated. This value can be no less than 1 and no more than VOLUMEMETER_MAX_REFRESHES, which is equal to 256. The default is VOLUMEMETER_STD_REFRESHES, which is equal to 4.

See also: -€**refresh:, ± setRefresh:, ±refreshesPerNewPeakBubble:**

setStereo
- **setStereo**

Sets the MiscVolumeMeter to display in stereo. If locked onto a sound, the MiscVolumeMeter should ignore this.

See also: -€**mono, -€setMono, -€stereo, -€setMono:, -€setStereo:**

setStereo:

- **setStereo:***sender*

Sets the MiscVolumeMeter to display in stereo. If locked onto a sound, the MiscVolumeMeter should ignore this.

See also: -€**mono, -€setMono, -€stereo, -€setMono:, -€setStereo**

setSound:

- **setSound:***this_sound*

Locks the MiscVolumeMeter to *this_sound*. If *this_sound* is NULL, the MiscVolumeMeter is unlocked and free to listen to all sounds.

See also: -€**sound**

setToInput

- **setToInput**

Tells the MiscVolumeMeter to listen to recording (input), not playing. If the MiscVolumeMeter is locked onto a sound, this should be ignored.

See also: -€**isInput:, ± setToOutput, ± setToInput:, ± setToOutput:**

setToInput:

- **setToInput:***sender*

Tells the MiscVolumeMeter to listen to recording (input), not playing. If the MiscVolumeMeter is locked onto a sound, this should be ignored.

See also: -€**isInput:, ± setToOutput, ± setToInput, ± setToOutput:**

setToOutput
- **setToOutput**

Tells the MiscVolumeMeter to listen to playing (output), not recording. If the MiscVolumeMeter is locked onto a sound, this should be ignored.

See also: **± setToInput, ± isInput: ± setToInput:, ± setToOutput:**

setToOutput:
- **setToOutput:***sender*

Tells the MiscVolumeMeter to listen to playing (output), not recording. If the MiscVolumeMeter is locked onto a sound, this should be ignored.

See also: **± setToInput, ± isInput: ± setToInput:, ± setToOutput**

setValueGray:
- **setValueGray:**(float)*this_value*

Sets the gray of the MiscVolumeMeter's volume bar to *this_value,* which should be between 0 and 1, inclusive.

See also: -€**setBackgroundGray:**, -€**setBubbleGray:**, -€**backgroundGray**, -€**valueGray**, -€**bubbleGray**

stop
- **stop**

Stops or suspends the MiscVolumeMeter.

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

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

Stops or suspends the MiscVolumeMeter.

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

sound
- **sound**

Returns the sound the MiscVolumeMeter has been locked on, or NULL if none.

See also: -€**setSound:**

valueGray:
- (float) **valueGray**

Returns the gray of the MiscVolumeMeter's volume bar.

See also: -€**setValueGray:**, -€**setBubbleGray:**, -€**setBackgroundGray**, -€**backgroundGray**, -€**bubbleGray**

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

Archives the MiscVolumeMeter to *stream*.

See also: -€**read:**

windowDidBecomeKey:
- **windowDidBecomeKey:***sender*

Runs or resumes the meter. This method should be called only by the MiscVolumeMeter's parent window if the MiscVolumeMeter has been set as the delegate of its window. See the introduction for more explanation.

See also: -€**windowDidBecomeMain:**, **±windowDidDeminiaturize:**, **±windowDidMiniaturize:**, **±windowWillClose:**

windowDidBecomeMain:
- **windowDidBecomeMain:***sender*

Runs or resumes the meter. This method should be called only by the MiscVolumeMeter's parent window if the MiscVolumeMeter has been set as the delegate of its window. See the introduction for more explanation.

See also: -€**windowDidBecomeKey:, ±windowDidDeminiaturize:, ±windowDidMiniaturize:, ±windowWillClose:**

windowDidDeminiaturize:

- **windowDidDeminiaturize:***sender*

Runs or resumes the meter. This method should be called only by the MiscVolumeMeter's parent window if the MiscVolumeMeter has been set as the delegate of its window. See the introduction for more explanation.

See also: -€**windowDidBecomeMain:, ±windowDidBecomeKey:, ±windowDidMiniaturize:, ±windowWillClose:**

windowDidMiniaturize:

- **windowDidMiniaturize:***sender*

Stops or suspends the meter. This method should be called only by the MiscVolumeMeter's parent window if the MiscVolumeMeter has been set as the delegate of its window. See the introduction for more explanation.

See also: -€**windowDidBecomeMain:, ±windowDidDeminiaturize:, ±windowDidBecomeKey:, ±windowWillClose:**

windowWillClose:

- **windowWillClose:***sender*

Stops or suspends the meter. This method should be called only by the MiscVolumeMeter's parent window if the MiscVolumeMeter has been set as the delegate of its window. See the introduction for more explanation.

See also: -€**windowDidBecomeMain:, ±windowDidDeminiaturize:, ±windowDidMiniaturize:, ±windowDidBecomeKey:**

Delegate Methods

meterDidUpdate:
- **meterDidUpdate:***sender*

Indicates that the meter has redrawn itself. **meterDidUpdate:** is called after the meter has already locked focus on itself, but after drawing has occured and before the meter has unlocked focus.

See also: -€**meterWillUpdate:, ± meterWillUpdateOnOwn:**

meterWillUpdate:
- **meterWillUpdate:***sender*

Indicates that the meter is about to redraw itself. **meterWillUpdate:** is called after the meter has already locked focus on itself.

See also: -€**meterWillUpdateOnOwn:, ± meterDidUpdate**

meterWillUpdateOnOwn:

- **meterWillUpdateOnOwn:***sender*

Indicates that the meter will update itself to reflect new sound information. This method is not called when the meter is simply redisplaying itself because of some external **display:** call, but *only* when the meter is internally updating itself while running. **meterWillUpdateOnOwn:** is called prior to the meter telling itself to display, and before the meter locks focus on itself. This method is always followed by **meterWillUpdate:,** though **meterWillUpdate:** is not always preceded by **meterWillUpdateOnOwn:.**

See also: -€**meterWillUpdate:, ± meterDidUpdate**