

Module

Inherits From: Object
Declared In: "Module.h"

Class Description

Module is the class from which all Resound Modules must descend. It provides the necessary instance variables and methods to dynamically load into Resound.

You should override **init** to set up *TheModuleMenuNode*. Be certain to begin your **init** method with **[super init];** to give Resound a chance to allocate and initialize *TheModuleMenuNode* prior to your making modifications to it. After **init** is called, Resound will ask your module for its main menu node, and use the information stored in the node and its subnodes to build submenus in Resound's Modules menu.

You shouldn't free *TheModuleMenuNode* or any *ModuleMenuNodes* you attach to it—they're freed automatically when Module is freed.

When you need to ask Resound to provide you with information, or need to inform Resound of something, you call upon

TheModuleController, the gateway object between your module and Resound's main application code. Module automatically allocates and initializes TheModuleController for you, so you don't have to. TheModuleController adheres to the ModuleProtocol and is discussed elsewhere.

Module declares five overridable delegate-like methods which you can use to change your module to reflect current conditions in the application. The default for each method simply returns self. Be sparing with each of these—they're very inefficient and should only be used if absolutely necessary.

Instance Variables

id <ModuleProtocol>	TheModuleController;
id	TheModuleMenuNode;

TheModuleController	The gateway object between your module and Resound.
---------------------	---

TheModuleMenuNode	The node object informing Resound how you'd like your menus set up.
-------------------	---

Method Types

Setup	- init ± free - setModuleControllerTo: - getModuleMenuNode:
-------	--

Optional overridable methods	± soundDidChange ± nowPlaying ± nowRecording ± didPlay ± didRecord
------------------------------	--

Instance Methods

didPlay

- **didPlay**

A sound has just finished playing. Override this method if you need to do special work when this occurs.

Warning: This method may be called twice for the same event.

didRecord

- **didRecord**

A sound has just finished recording. Override this method if you need to do special work when this occurs.

Warning: This method may be called twice for the same event.

free

- **free**

Frees the Module, TheModuleMenuNode, and all subnodes of TheModuleMenuNode.

getModuleMenuNode:

- **getModuleMenuNode**

Returns TheModuleMenuNode. Resound will call this after calling init, so you need to have your node set up by then.

init

- **init**

Initializes the module. You should override this to at least set up TheModuleMenuNode as you like.

nowPlaying

- **nowPlaying**

A sound has begun playing. Override this method if you need to do special work when this occurs. *Warning:* This method may be called twice for the same event.

nowRecording

- **nowRecording**

A sound has begun recording. Override this method if you need to do special work when this occurs. *Warning:* This method may be called twice for the same event.

setModuleControllerTo:

- **setModuleControllerTo:***theModuleController*

Sets TheModuleController. You should never call this method. It is called by Resound to prepare the module when being dynamically loaded.

soundDidChange

- **soundDidChange**

Informs the Module that Resound's current Sound, SoundView, or Window has changed. This method is very

costly, so only override it if you absolutely must. This method is called whenever a user edits a sound or modifies it, switches to another sound, closes a sound or creates a new one, miniaturizes a sound window, etc. Essentially this method is called whenever an inspector might need to be updated as to the current status of a sound. There is one exception, however: if the user has just changed the selection or view size, or scrolled the window, **soundDidChange** will not be called. It's just too costly. It is possible that the sound changed so that there is *no* current sound or soundview (if the user closed all the windows, for example).

At all cost, do not use this method to modify the sound that's been changed, as this will issue yet *another* **soundDidChange** message, and you'll be stuck in a recursive loop. *Warning:* This method may be called twice for the same event.