MiscMailSpeaker

Inherits From: Speaker

Declared In: <misckit/MiscMailSpeaker.h>

Class Description

Note: It is recommended that you use the MiscMailApp object to communicate with NeXT's Mail.app (and its replacements) rather than this object. This object depends upon a private API which may not always be available. This object, in other words, is in more danger of becoming obsolete than MiscMailApp.

The MiscMailSpeaker is a Speaker subclass which allows access to the API of NeXT's Mail.app program. By using this object, you can open and deliver ªComposeº windows. You can also set the Subject:, To:, and Cc: fields as well as the message body to arbitrary string values. The easiest (and strongly *not* recommended) method of use is to use ±**openSend** to get a compose window, ±**setSubject:**, ±**setTo:**, ±**setCc**, and ±**setBody:** to configure the message and ±**deliver** to send the message off. Before you can actually send messages to the MiscMailSpeaker, however, it is important that you first connect it to a Mach port. One way to do this is shown in the sample code below.

There are really two parallel APIs represented by this object. The simplest API uses the ªcurrentº Compose window. There are a few disadvantages to this: although you can open a new compose window, it will not

necessarily become the ªcurrentº window. As a result, subsequent messages to set various fields in the message could clobber an existing, partially composed message in Mail.app. Also, some Mail.app replacements do not support this particular API. The better way to go is to use the ±**openSend:** method, which returns an id to represent the newly opened compose window. All messages which set fields in the Compose window have ±**xxx:InWindow:** counterparts which use the window id returned by ±**openSend:** to place the text in the proper Compose window. Use ±**deliver:** to deliver a window specified by a certain id. This is the API of choiceÐif a MiscMailSpeaker must be used, please use this interface for the best results with future versions of Mail.app and Mail.app replacements.

A typical use of this class would be to allow customers an easy way to provide product feedback. For example, here is a method which demonstrates the use of a MiscMailSpeaker to open and fill a compose window with optional delivery:

```
- sendMailTo:(const char *)to subject:(const char *)subject
          body:(const char*)body andDeliver:(BOOL)flag
{
     port_t mail;
     id speaker;
     int theWindow = 0; // Initialization is not really necessary.

     speaker = [MiscMailSpeaker new];
     mail = NXPortFromName("MailSendDemo", "");

     [speaker setSendPort:mail];
     [speaker openSend:&theWindow];
     if (subject)[speaker setSubject:(char *)subject inWindow:theWindow];
     if (to)[speaker setTo:(char *)to inWindow:theWindow];
     if (body)[speaker setBody:(char *)body inWindow:theWindow];
     if (flag)[speaker deliver:theWindow];
     [speaker free];
     return self;
}
```

Note that the above code attempts to connect to an already running mail application but doesn't attempt to launch it. This object is used by the MiscInfoController class to send mail messages for product feedback. It is

also used as a part of the MiscMailApp object. In fact, using MiscMailApp provides the above method as well as others and is even easier to use than this object. As a final note, this object was machine generated from MiscMail.msg using the *msgwrap* program provided by NeXT. Here are the contents of that file:

```
- openSend;
- setTo:(char *)toField;
- setSubject:(char *)subjectField;
- setCc:(char *)ccField;
- setBody:(char *)bodyField;
- deliver;
- openSend:(int *)sender;
- setTo:(char *)toField inWindow:(int)sender;
- setSubject:(char *)subjectField inWindow:(int)sender;
- setCc:(char *)ccField inWindow:(int)sender;
- setBody:(char *)bodyField inWindow:(int)sender;
- deliver:(int)sender;
```

The last six messages represent the preferred API for messaging Mail.app.

Method Types

Manipulating Mail.app Compose windows
- deliver
- deliver:
- openSend
- openSend:
- setBody:
- setBody:inWindow:
- setCc:
- setCc:inWindow:
- setSubject:
- setSubject:inWindow:

- setTo:
- setTo:inWindow:

Instance Methods

deliver

- (int)**deliver**

Directs Mail.app to deliver the ªcurrentº Compose window. Returns **self**.

See also: ± deliver:

deliver:

- (int)**deliver:**(int)*sender*

Directs Mail.app to deliver the Compose window identified by *sender*. Returns **self**.

See also: ± deliver

openSend

- (int)**openSend**

Opens a new Compose window in Mail.app. Warning: This is not guaranteed to make the new window the ªcurrentº window. Returns **self**.

See also: ± openSend:

openSend:

- (int)**openSend:**(int *)*sender*

Opens a new Compose window in Mail.app. The id of the created window is returned by reference in *sender*. Returns **self**.

See also: ± openSend

setBody:

- (int)**setBody:**(const char *)*bodyField*

Sets the body of the message in the current Compose window to the text in *bodyField*. Returns **self**.

See also: ± setBody:inWindow:

setBody:inWindow:

- (int)**setBody:**(const char *)*bodyField*
  inWindow:(int)*sender*

Sets the body of the message in the Compose window identified by *sender* to the text in *bodyField*. Returns **self**.

See also: ± setBody:

setCc:

- (int)**setCc:**(const char *)*ccField*

Sets the Cc: field of the message in the current Compose window to the text in *ccField*. Returns **self**.

See also: ± setCc:inWindow:

setCc:inWindow:

- (int)**setCc:**(const char *)*ccField*
  inWindow:(int)*sender*

Sets the Cc: field of the message in the Compose window identified by *sender* to the text in *ccField*. Returns **self**.

See also: ± setCc:

setSubject:

- (int)**setSubject:**(const char *)*subjectField*

Sets the Subject: field of the message in the current Compose window to the text in *subjectField*. Returns **self**.

See also: ± setSubject:inWindow:

setSubject:inWindow:

- (int)**setSubject:**(const char *)*subjectField*
  inWindow:(int)*sender*

Sets the Subject: field of the message in the Compose window identified by *sender* to the text in *subjectField*. Returns **self**.

See also: ± setSubject:

setTo:

- (int)**setTo:**(const char *)*toField*

Sets the To: field of the message in the current Compose window to the text in *toField*. Returns **self**.

See also: ± setTo:inWindow:

setTo:inWindow:

- (int)**setTo:**(const char *)*toField*
  inWindow:(int)*sender*

Sets the To: field of the message in the Compose window identified by *sender* to the text in *toField*. Returns **self**.

See also: ± setTo: