SoundEditor

This programming example illustrates how easy it is to use the three SoundKit classes Ð Sound, SoundView, and SoundMeter. Multiple sounds can be open at the same time, with full cut, copy and paste functionality, using the mouse to select portions of the sound waveform. A control panel, consisting of a SoundMeter and Play, Stop, Pause,' Record Buttons, operates on whichever sound is in the current key window.

The **SoundDocument** class manages a single window containing a **ScrollingSound**, which is simply a subclass of ScrollView that contains a SoundView. The ScrollingSound is contained in SoundDocument.nib, which is loaded for each new window. As with most of the NeXT programming examples, it's as instructive to explore the .nib files as to read the source code. Note that this .nib file's owner is of class SoundDocument.

The **SoundController** class manages communication between the SoundDocuments and the rest of the interface (the menus and the control panel).

Every time the user requests a new window with the New or Open... commands, the SoundController creates a new instance of SoundDocument. SoundDocument in its **init** method then loads in a new SoundDocument.nib, which displays the window with the ScrollingSound. To create the SoundMeter in the control panel, we created a custom view, dragged the icon of /usr/include/soundkit/SoundMeter.h from Workspace into IB's Classes suitcase, and used the Inspector to change the view's class.

You can use this simple example as a basis for many useful extensions Ð for example, zooming, mixing, filtering, changing sound format, etc. If you're interested in a more full-function editor, there is at least one commercial product (SoundWorks™ from Metaresearch), as well as some public-domain editors available with source code on the Internet archive servers.

Changes for Release 3.0

Features have been added to illustrate use of the new Audio Transform Compression (ATC) format, various new supported format conversions, and

miscellaneous new API in the Sound Kit.

A **SaveTo** panel has been added which supports

(1) changing the data format on save to linear, mu-law, or ATC compressed,
(2) converting mono to stereo, or stereo to mono, and
(3) converting sampling rates among the three commonly used on NeXT computers.

Many other format and sampling-rate conversions are now supported by the Sound object's upgraded **convertToFormat:**... method (see the Sound release notes and on-line documentation for details); the cases illustrated here are only the most generall(eful. Compression and decompression are carried out using the DSP, so they operate faster than real time for all standard sampling rates. Sampling-rate conversion does not use the DSP, so it is typically slower than real time.

In the **SoundDocument** class, if the soundfile format cannot be displayed by

SoundView, (e.g. because it is compressed), the format is converted to LINEAR_16 on input. This is detected by the SoundView object returning **nil** in response to its **setSound:** method when the sound is not displayable. (Empty sounds are defined as displayable in this context. To be undisplayable, the sound must contain data in a format not supported by SoundView.)

The new SoundView method **isPlayable** is illustrated. When the play button is pressed, the current sound is send the isPlayable message. If the method returns nil, the system beep is played instead of attempting to play the sound.