ErrorSound

ErrorSound Set up to use non-standard sounds for alerts

#include <Dialogs.h> **Dialog Manager**

```
void        ErrorSound(soundProc );
ProcPtr     soundProc ;     routine to generate alert beeps; NIL=disable
```

ErrorSound lets you override the standard sounds that are made when alerts are invoked.

soundProc is the address of a pascal-style procedure. This routine will get control at each stage of each alert. A value of NIL disables alert beeping altogether and also disables the menu bar-blinking that occurs when the speaker volume has been set to 0.

Returns: none

Notes: If you never call this function, the alert will emit simple beeps - up to 3; one beep for the current alert stage - at the current speaker volume (adjustable via the control panel DA). In the event that the volume has been set to 0, the standard "sound" is a flashing of the menu bar. See **SysBeep**.

If you call **ErrorSound**(0), beeping and flashing will not occur.

To customize the sounds, use **ErrorSound**(mySounds), as illustrated in the following example.

Note: Sound number 1 is the sound made when a user clicks outside of a modal dialog box (as well as in a stage-1 alert).

Example

```
#include <Dialogs.h>
#include <Sound.h>

pascal void MySoundProc( short sndNum );

#define kSndResNum  128                    /* 'snd ' resource number */

pascal void MySoundProc(short sndNum)
/* sndNum will range from 0 to 3 */
{
    SndChannelPtr myChan = 0L;
    Handle mySound;
    OSErr err;

    if (sndNum == 0) return;
    mySound = GetResource( soundListRsrc, kSndResNum );
    err = SndNewChannel( &myChan, 0, 0, 0L );
    HLock( mySound );
    err = SndPlay( myChan, mySound, FALSE );
    HUnlock( mySound );
    err = SndDisposeChannel( myChan, FALSE );
}
```

```
ErrorSound( MySoundProc );        /* use custom sounds in next alert */
```