Windows Sound Hack Utility
Version 1.0 by Jerry Joplin, 1991

List of available topics:

Introduction

SndHack! (?)   What is SndHack?   Why is SndHack a <u>hack?</u>

SndHack is a Windows Sound Function Interpreter.   It
interprets text as a series of calls to Windows
<u>Sound Functions</u> .

Each command consists of a function name and a list
of zero or more parameters enclosed in parenthesis.

EXAMP001.WSN
        ; SndHack processes a text file 1 line at a time.
        ; Empty lines are ignored, and all characters following
        ; a semicolon are ignored.
        ; Many examples start with a CloseSound().   This
        ; will just make sure that we have closed the
        ; sound device from any previous OpenSound()s.
        ;we have done.
        CloseSound();
        OpenSound();
        SetVoiceNote(1, C4, 4, 0);
        StartSound();
        WaitSoundState(S_QUEUEEMPTY);
        CloseSound();

SndHack is designed for the developer but can be used by a
casual user.   For developers: have you ever noticed the
documentation for the Windows <u>Sound Functions</u> is
really lacking?   You end up having to <u>hack</u> your way
through the interface anyway so there might as well be a tool
designed specifically for this job.

Sound Functions

File Menu

SndHack interpreted programs may be saved and retrieved
in files.   The default extension for these files is .WSN
and may be manipulated using the following standard
menu commands:

New
Open
Save
Save As

Editing Files

SndHack uses a standard Edit window for its text editor.
List of available edit menu commands and their accelerators:

Undo   - Alt+BkSp
Cut   - Shift+Del
Copy   - Ctrl+Ins
Paste   - Shift+Ins
Delete   - Del

SndHack also has a Paste Function command to allow for dialog box guided Sound Function entry.

Play

The Play menu command initiates "playing" of the current edit buffer.

CloseSound

Usage:
CloseSound();

Parameters:
None

Description:
Closes the sound device.   It also flushes all voice
queues and frees any memory associated with the
voice queues.

See Also:
OpenSound
SetVoiceQueueSize

EXAMP002.WSN
;**BEWARE**
;The CloseSound function will flush the voice queues, which
;will cause any left over queued sounds to be lost.   In this
;example the note queued in the SetVoiceNote call will not
;be played.
CloseSound();
OpenSound();
SetVoiceNote(1, C4, 4, 0);
StartSound();
CloseSound();

CountVoiceNotes

Usage:
    CountVoiceNotes(Voice);

Parameters:
    Voice:
        Integer containing the voice queue number.
        This will almost always be 1, unless there
        is a special multi-voice sound driver installed.

Description:
    Counts the number of voice notes in the specified
    voice queue.   The call to the function itself is
    supported in SndHack.   However, currently there
    is no way to retrieve the value obtained in the call,
    so this function isn't much use in SndHack.

See Also:
    SetVoiceNote

GetThresholdEvent

Usage:
GetThresholdEvent();

Parameters:
None

Description:
Retrieves a pointer to a threshold event.
The call to the function itself is supported
in SndHack.   However, currently there is no way
to use the pointer obtained in the call, so this
function isn't much use in SndHack.

See Also:
GetThresholdStatus
SetVoiceThreshold

GetThresholdStatus

Usage:
GetThresholdStatus();

Parameters:
None

Description:
Returns a threshold event status mask.
The call to the function itself is supported in
SndHack.   However, currently there is no way to
use the mask obtained in the call, so this
function isn't much use in SndHack.

See Also:
GetThresholdEvent
SetVoiceThreshold

OpenSound

Usage:
OpenSound();

Parameters:
None

Description:
Opens the sound device for exclusive use by
SndHack.   This function may fail if another
application is using the sound device, or if
it has already been opened in SndHack.

See Also:
CloseSound

EXAMP003.WSN
;OpenSound() fails if the sound device is already
;in use by another application, or SndHack.
OpenSound();
OpenSound();   This command fails!

SetSoundNoise

Usage:
SetSoundNoise(Source, Duration);

Parameters:
Source:
Specifies the noise source.
Can be one of the following:
S_PERIOD512, SourceFreq = TargetFreq/512 (high pitch)
S_PERIOD1024, SourceFreq = TargetFreq/1024
S_PERIOD2048, SourceFreq = TargetFreq/2048 (low pitch)
S_PERIODVOICE, SourceFreq is from voice channel 3.
S_WHITE512,   SourceFreq = TargetFreq/512 (high pitch)
S_WHITE1024,   SourceFreq = TargetFreq/1024
S_WHITE2048,   SourceFreq = TargetFreq/2048 (low pitch)
S_WHITEVOICE, SourceFreq is from voice channel 3.
Duration:
Integer containing duration of the noise
in clock tics.

Description:
Sets the Source and Duration of a noise in
the sound device. This function seems to have
no effect when used with the default sound driver.

See Also:
SetVoiceSound

EXAMP004.WSN
;The SetSoundNoise function seems to have
;no effect on the default sound driver.
CloseSound();
OpenSound();
SetSoundNoise(S_PERIOD512, 10);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
SetSoundNoise(S_PERIOD2048, 10);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
SetSoundNoise(S_PERIODVOICE, 10);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
SetSoundNoise(S_WHITEVOICE, 20);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);

```
StartSound();
WaitSoundState(S_QUEUEEMPTY);
CloseSound();
```

SetVoiceAccent

Usage:
SetVoiceAccent(Voice, Tempo, Volume, Mode, Pitch);

Parameters:
Voice:
Integer containing the voice queue number.
This will almost always be 1, unless there
is a special multi-voice sound driver installed.
Tempo:
Integer containing the number of quarter notes
played per minute.   Valid values are 32 - 255.
Volume:
Integer containing the volume.   Valid values
are 0 - 255.
Mode:
Specifies the mode for playing notes.
Can be one of the following:
S_LEGATO, notes play for the full length of
        the note and blend with next note.
S_NORMAL, notes play for the full length of
        the note, but stop before the next note.
S_STACCATO, notes do not play for the full lenth
        of the note and stop before the next note.
Pitch:
Integer containing value to add to each note
before it is played.   Note values are numbered
1 - 84 for 12 notes in 7 octaves.   If resulting
notes extend past 84, then they wrap back to the
beginning.

Description:
Places an accent definition in a voice queue.
The definition does not count as a note in the
voice queue, as retrieved by CountVoiceNotes.
However it does use space in the voice queue,
and can cause queue full errors. This function
seems to have no effect when used with the
default sound driver.

See Also:
SetVoiceNote

EXAMP005.WSN
;The SetVoiceAccent function seems to have

```
;no effect on the default sound driver.
CloseSound();
OpenSound();
SetVoiceAccent(1, 120, 64, S_NORMAL, 0);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,D3,8,0);
SetVoiceNote(1,E3,8,0);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,E3,8,0);
SetVoiceNote(1,D3,8,0);
SetVoiceNote(1,0,8,0);
StartSound();
WaitSoundState(S_QUEUEEMPTY);
SetVoiceAccent(1, 200, 32, S_LEGATO,12);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,D3,8,0);
SetVoiceNote(1,E3,8,0);
SetVoiceNote(1,C3,8,0);
SetVoiceNote(1,E3,8,0);
SetVoiceNote(1,D3,8,0);
SetVoiceNote(1,0,8,0);
StartSound();
WaitSoundState(S_QUEUEEMPTY);
CloseSound();
```

SetVoiceEnvelope

Usage:
SetVoiceEnvelope(Voice, Shape, Repeat);

Parameters:
Voice:
Integer containing the voice queue number.
This will almost always be 1, unless there
is a special multi-voice sound driver installed.
Shape:
Integer containing an index into an OEM
supplied waveshape table.   Unfortunately
this table is not supplied with the default
sound driver.
Repeat:
Integer containing the repeat count of the
waveshape during the playing of individual notes.

Description:
Places an envelope definition in a voice queue.
The definition does not count as a note in the voice
queue, as retrieved by CountVoiceNotes.
However it does use space in the voice queue, and
can cause queue full errors.   This function seems
to have no effect when used with the default
Windows default sound driver.

See Also:
SetVoiceNote

EXAMP006.WSN
;The SetVoiceEnvelope function seems to have
;no effect on the default sound driver.
CloseSound();
OpenSound();
SetVoiceEnvelope(1, 0, 1);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
SetVoiceEnvelope(1, 1, 10);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
SetVoiceEnvelope(1, 2, 20);
SetVoiceNote(1, 50, 4, 0);
SetVoiceSound(1, 0x01000001, 20);
StartSound();

```
WaitSoundState(S_QUEUEEMPTY);
CloseSound();
```

SetVoiceNote

Usage:
    SetVoiceNote(Voice, Value, Length, Cdots);

Parameters:
    Voice:
        Integer containing the voice queue number.
        This will almost always be 1, unless there
        is a special multi-voice sound driver installed.
    Value:
        Specifies the note that is to be played.
        This is normally an integer from 0 - 84,
        with 0 being a rest, and 1 - 84 corresponding
        to the 12 notes in 7 octaves.   SndHack allows
        notes to be specified as the name of the note
        with the octave appended to the end.   Sharps are
        specified by # or + and flats by -.
        Example: C4, C#1, G-3.
    Length:
        Integer containing the inverse of the note
        length, e.g. a length of 2 specifies a 1/2 note,
        and a length of 8 specifies a 1/8 note.
    Cdots:
        Integer containing the number of 'dots' to
        be placed on the note.   Each dot extends the
        duration of the note by 1/2 of its specified
        length.   (The Cdots parameter seems to be
        ignored with the default sound driver.)
Description:
    Queues a note into a voice queue.

See Also:
    CountVoiceNotes

EXAMP007.WSN
    ; SndHack allows notes to be specified as the
    ; name of the note with the octave appended to
    ; the end.   Sharps are specified by # or +
    ; and flats by -.
    CloseSound();
    OpenSound();

    SetVoiceNote(1, 1, 4, 0);   This note and the next are equivalent.
    SetVoiceNote(1, C1, 4, 0);

```
        SetVoiceNote(1, 13, 4, 0);     This note and the next are
equivalent.
        SetVoiceNote(1, C2, 4, 0);

        SetVoiceNote(1, C4, 4, 0);      Plays C in Octave 4
        SetVoiceNote(1, C#1, 4, 0);   Plays C sharp in Octave 1
        SetVoiceNote(1, G-3, 4, 0);    Plays G flat in Octave 3

        StartSound();
        WaitSoundState(S_QUEUEEMPTY);
        CloseSound();
```

SetVoiceQueueSize

Usage:
    SetVoiceQueueSize(Voice, QueueSize);

Parameters:
    Voice:
        Integer containing the voice queue number.
        This will almost always be 1, unless there
        is a special multi-voice sound driver installed.
    QueueSize:
        Specifies the size of the voice queue in
        bytes which the sound driver should attempt
        to allocate.   The size does not apply across
        calls to CloseSound   which
        deallocates all memory associated with voice queues.

Description:
    Sets the size of a voice queue.

See Also:
    CloseSound

EXAMP008.WSN
    ; Be careful when setting the voice queue
    ; size.   If it is set too small, subsequent
    ; requests to queue items to a voice queue
    ; will fail.
    CloseSound();
    OpenSound();
    SetVoiceQueueSize(1, 10);   10 bytes is too small!!
    SetVoiceNote(1, C4, 4, 0);
    SetVoiceNote(1, C1, 4, 0);   This request will fail!
    SetVoiceNote(1, C1, 8, 0);
    SetVoiceNote(1, C1, 4, 0);
    StartSound();
    WaitSoundState(S_QUEUEEMPTY);
    CloseSound();

SetVoiceSound

Usage:
> SetVoiceSound(Voice, Frequency, Duration);

Parameters:
> Voice:
>> Integer containing the voice queue number.
>> This will almost always be 1, unless there
>> is a special multi-voice sound driver installed.
>
> Frequency:
>> A long integer that specifies the frequency
>> in Hertz in the high order word, and a fraction
>> in the low order word.
>> Examples:
>>> 0x01000000 specifies 256.0 Hertz.
>>> 0x01000001 specifies 256.1 Hertz.
>>> 0x02000001 specifies 512.1 Hertz.
>
> Duration:
>> Integer containing duration of the sound in
>> clock tics.

Description:
> Queues a sound into a voice queue.   The sound
> does not count as a note in the voice queue,
> as retrieved by CountVoiceNotes.
> However it does use space in the voice queue,
> and can cause queue full errors.

See Also:
> CountVoiceNotes

EXAMP009.WSN
```
;The SetVoiceSound function plays a
;sound specified by a frequency and
;length in clock tics.
CloseSound();
OpenSound();
SetVoiceSound(1, 0x01000000, 8);   Freq : 0x0100 Hz
SetVoiceSound(1, 0x01200000, 8);   Freq : 0x0200 Hz
SetVoiceSound(1, 0x01400000, 8);   Freq : 0x0400 Hz
SetVoiceSound(1, 0x01800000, 8);   Freq : 0x0800 Hz

SetVoiceSound(1, 0x01000000, 8);   Repeat the pattern
SetVoiceSound(1, 0x01200000, 8);
SetVoiceSound(1, 0x01400000, 8);
```

```
SetVoiceSound(1, 0x01800000, 8);

SetVoiceSound(1, 0x01000000, 8);   One more time
SetVoiceSound(1, 0x01200000, 8);
SetVoiceSound(1, 0x01400000, 8);
SetVoiceSound(1, 0x01800000, 8);

StartSound();
WaitSoundState(S_QUEUEEMPTY);
CloseSound();
```

SetVoiceThreshold

Usage:
SetVoiceThreshold(Voice, Notes);

Parameters:
Voice:
Integer containing the voice queue number.
This will almost always be 1, unless there
is a special multi-voice sound driver installed.
Notes:
Integer that defines the number of notes of
the threshold level in the voice queue.

Description:
Sets the threshold level for a voice queue.
This greatly effects the operation of the
WaitSoundState function.
WaitSoundState(S_THRESHOLD) or
WaitSoundState(S_ALLTHRESHOLD)
wait until a threshold level is crossed in
a voice queue.   If the threshold level is
greater than the number of notes placed in
the queue, the threshold level will never be
crossed and the WaitSoundState will hang
forever and the machine has to be rebooted.

See Also:
GetThresholdStatus
SetVoiceThreshold
WaitSoundState

EXAMP010.WSN
;The call to SetVoiceThreshold sets the
;threshold level to 4 notes so more
;notes can be placed on the voice queue
;while the sound is still playing.

CloseSound();
OpenSound();

SetVoiceThreshold(1, 4);
SetVoiceNote(1,C3,8,1);
SetVoiceNote(1,D3,16,0);
SetVoiceNote(1,C3,8,0);

```
        SetVoiceNote(1,B-2,8,0);
        SetVoiceNote(1,A2,8,0);
        SetVoiceNote(1,B-2,8,0);
        SetVoiceNote(1,C3,4,0);
        StartSound();
        WaitSoundState(S_THRESHOLD);
        SetVoiceNote(1,G2,8,0);
        SetVoiceNote(1,A2,8,0);
        SetVoiceNote(1,B-2,4,0);
        SetVoiceNote(1,A2,8,0);
        SetVoiceNote(1,B-2,8,0);
        SetVoiceNote(1,C3,4,0);
        WaitSoundState(S_QUEUEEMPTY);
        CloseSound();
```

EXAMP011.WSN

```
        ;The call to SetVoiceThreshold sets the
        ;threshold level to 2 notes in this example.
        ;This causes the last notes in the voice
        ;queue to be lost since the last call to
        ;CloseSound() will flush the voice queue.

        CloseSound();
        OpenSound();

        ;Warning: Do not set the threshold level
        ;to 4 or greater in this example.   This
        ;will cause the WaitSoundState to hang
        ;forever.

        SetVoiceThreshold(1, 2);
        SetVoiceNote(1, 1, 4, 0);
        SetVoiceNote(1, 13, 4, 0);
        SetVoiceNote(1, 25, 4, 0);
        SetVoiceNote(1, 37, 4, 0);

        StartSound();
        WaitSoundState(S_THRESHOLD);
        CloseSound();
```

StartSound

Usage:
StartSound();

Parameters:
None

Description:
Starts playing of queued sound commands.

See Also:
StopSound

StopSound

Usage:
StopSound();

Parameters:
None

Description:
Stops the playing of queued sound commands.
This function also flushes all of the voice
queues.

See Also:
StartSound

SyncAllVoices

Usage:
SyncAllVoices();

Parameters:
None

Description:
Places a sync mark in all voice queues.   This will
insure that all voice queues are in sync.   This
function isn't much good with the default sound
driver since there is only 1 voice available.

See Also:
StartSound

WaitSoundState

Usage:
WaitSoundState(State);

Parameters:
State:
Specifies the sound state to wait for.
Can be one of the following:
S_ALLTHRESHOLD, threshold level for all voice queues
S_QUEUEEMPTY, voice queues are empty
S_THRESHOLD, threshold level for a single voice queue

Description:
Waits for a specified sound event to occur.   Be
careful with this function when setting values
for the threshold level with SetVoiceThreshold.
SndHack changes the shape of the cursor to 2 connected
notes during the execution of WaitSoundState.

See Also:
SetVoiceThreshold

Hack (in the best sense) this means to 'spelunk'
  or to explore areas of computer architecture.