Paper 048-29

Making Music in SAS â : Using Sound to Alert Users of Errors and Data Discrepancies

David Fielding, AAI Development Service, Mississauga, ON

ABSTRACT

Did you know you could have SAS play music? This paper explores the SOUND and SLEEP routines. Let SAS alert you of an error in your program, a specific type of data, or just add some excitement in your life.

INTRODUCTION

In today's fast pace environment it is important to be alerted to problems immediately. We will look at how we can use sound in our programs to alert us during runtime.

We will start with an introduction into the physics of music and how this relates to the CALL SOUND routine. Then we will look at CALL SLEEP. Then we will look at a macro for translating sheet music into SAS code using CALL SOUND and CALL SLEEP.

MUSIC BACKGROUND

Sound waves each have a specific frequency; it is the frequency that distinguishes the notes we hear. The frequency is measured in hertz which measures cycles per second. The chart below shows how each note relates to the frequency.
Table 1. Notes and related frequencies. (Wenzel)

Notes	Frequency (octaves)				
A	55.00	110.00	220.00	440.00	880.00
A#/Bb	58.27	116.54	233.08	466.16	932.32
B	61.74	123.48	246.96	493.92	987.84
C	65.41	130.82	261.64	523.28	1046.56
C#/Db	69.30	138.60	277.20	554.40	1108.80
D	73.42	146.84	293.68	587.36	1174.72
D#/Eb	77.78	155.56	311.12	622.24	1244.48
E	82.41	164.82	329.64	659.28	1318.56
F	87.31	174.62	349.24	698.48	1396.96
F#/Gb	92.50	185.00	370.00	740.00	1480.00
G	98.00	196.00	392.00	784.00	1568.00
G#/Ab	103.83	207.66	415.32	830.64	1661.28

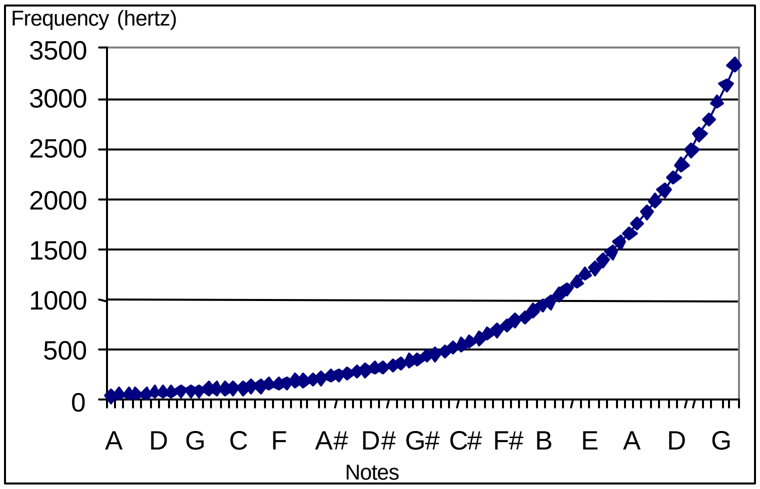

Figure 1. Higher notes increase exponentially.

CALL SOUND

CALL SOUND(*frequency*,*duration*) where the *frequency* specifies the sound frequency in terms of cycles per second and the *duration* specifies the sound duration in milliseconds. (SAS Help)

APPLICATION

The following sample dataset is used for the CALL SOUND examples. There are 5 observations for the variable n (0,3,5,.,6) which will be used in the examples.

```
data sample;
  input n;
cards;
0
3
5
.
6
;
run;
```

EXAMPLE 1

If you batch submit programs on a regular basis and would like to be notified of an error in a data step before checking the log. Add the statement if _ERROR_=1 and use CALL SOUND. Of course a dataset that produces a lot of errors may not go over well with nearby co-workers. In the example when calculating b=2/n, when n is 0 the error "Division by zero detected" occurs and _ERROR_ becomes 1. When _error_ is set to 1, the if statement then produces the sound and the user is immediately aware of a problem in the dataset. The notes E (659), D (587) and C (523) are played for 200 milliseconds.

```
data example1;
  set sample;
  b=2/n;
  if _error_=1 then do;
    call sound(659,200);
    call sound(587,200);
    call sound(523,200);
  end;
run;
```

EXAMPLE 2

In the pharmaceutical industry, medications and adverse events are coded to standard dictionaries for analysis. To prevent sending a client a term that hasn't been coded, just add code to notify of missing codes. In the example, values 0,3,5 are coded but . and 6 remain uncoded. When the program is run the sound notifies the user immediately of uncoded terms as they are read in. The notes C (523), D (587) and E (659) are played for 200 milliseconds.

```
data example2;
  set sample;
  length code $5.;
  if n in (0,3,5) then code="CODED";
  if code="" then do;
    call sound(440,200);
    call sound(523,200);
    call sound(659,200);
  end;
run;
```

Sound can be used to notify the user of specific data that may require further action.

Another use is to communicate to yourself in a program. If you've changed a program and wanted to make sure you validated it before running it next time, set up a call sound that can be turned off when validation is complete.

Of course you can always just add a song to any routinely run program just to add some excitement in your life or others.

CALL SLEEP

CALL SLEEP(*n*,*unit*) where **n** specifies the number of units of time that you want to suspend the execution of a DATA step. Specifies the unit of time, as a power of 10, which is applied to *n*. The default is 1 in a Windows or a PC environment. In other environments the default is .001. Where 1 corresponds to a second. (SAS Help)

EXAMPLE

```
data _null_;
  call sleep(30,1);
run;

NOTE: DATA statement used:
      real time           30.03 seconds
```

DF3 MACRO

The DF3 macro allows you to play a song by sending the musical note, its octave and how long to play it.

The notes are A-G as on any piano keyboard. To indicate a flat add ‘b’ or a sharp add ‘#’ (Middle C is octave 3). To play a rest use R with the length.

The length is how long to hold the note, in 4/4 timing 1 would be a quarter note, 2 a half note and 4 a whole note.

The duration of the call sound is supposed to be in milliseconds but the length of the sound is not consistent across PC’s. Adjust the macro variable *pc* as a multiplier to the length of a note. If your PC plays the sample song very slowly decrease this number.

```
%let pc=1.25;

%macro df3(note,octave,length);

select(&note.);

  when('A')  call sound(55*(2**&octave.),&length.*160*&pc.);
  when('A#') call sound(58*(2**&octave.),&length.*160*&pc.);
  when('Bb') call sound(58*(2**&octave.),&length.*160*&pc.);
  when('B')  call sound(62*(2**&octave.),&length.*160*&pc.);
  when('C')  call sound(65*(2**&octave.),&length.*160*&pc.);
  when('C#') call sound(69*(2**&octave.),&length.*160*&pc.);
  when('Db') call sound(69*(2**&octave.),&length.*160*&pc.);
  when('D')  call sound(73.5*(2**&octave.),&length.*160*&pc.);
  when('D#') call sound(73.5*(2**&octave.),&length.*160*&pc.);
  when('Eb') call sound(78*(2**&octave.),&length.*160*&pc.);
  when('E')  call sound(82*(2**&octave.),&length.*160*&pc.);
  when('F')  call sound(87*(2**&octave.),&length.*160*&pc.);
  when('F#') call sound(92.5*(2**&octave.),&length.*160*&pc.);
  when('Gb') call sound(92.5*(2**&octave.),&length.*160*&pc.);
  when('G')  call sound(98*(2**&octave.),&length.*160*&pc.);
  when('G#') call sound(104*(2**&octave.),&length.*160*&pc.);
  when('Ab') call sound(104*(2**&octave.),&length.*160*&pc.);

  when('R')  call sleep((&length./3)*&pc.,1);

otherwise;
end;

%mend;
```

/* SAMPLE */

```
/* Old MacDonald Had a Farm */

data _null_;

  do i=1 to 2:
    %df3('C',3,1);
    %df3('C',3,1);
    %df3('C',3,1);
    %df3('G',2,1);
    %df3('A',3,1);
    %df3('A',3,1);
    %df3('G',2,2);
    %df3('E',3,1);
    %df3('E',3,1);
    %df3('D',3,1);
    %df3('D',3,1);
    %df3('C',3,2);
    if i=1 then do;
      %df3('R',1,2);
      %df3('G',2,2);
    end;
  end;

  %df3('G',2,.5);
  %df3('G',2,.5);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('G',2,.5);
  %df3('G',2,.5);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('C',3,2);

  %df3('C',3,.5);
  %df3('C',3,.5);
  %df3('C',3,1);
  %df3('C',3,.5);
  %df3('C',3,.5);
  %df3('C',3,1);
  %df3('C',3,.5);
  %df3('C',3,.5);
  %df3('C',3,.5);
  %df3('C',3,.5);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('C',3,1);
  %df3('G',2,1);
  %df3('A',3,1);
  %df3('A',3,1);
  %df3('G',2,2);

  %df3('E',3,1);
  %df3('E',3,1);
  %df3('D',3,1);
  %df3('D',3,1);
  %df3('C',3,3);

run;
```

CONCLUSION

It is always important to review your log after running a SAS program, by adding CALL SOUND you will be notified of data discrepancies and errors during runtime. The DF3 macro can be used to translate sheet music into SAS music and let you entertain your friends.

REFERENCES

Wenzel, Charles. “Musical Notes”, *Techlib.com Reference Materials.* http://www.techlib.com/reference/musical_note_frequencies.htm (January 14, 2004).

SAS Institute Inc., “The SAS System for Windows, Release 8.02, SAS System Help”, *CALL SLEEP*, Cary, NC: SAS Institute Inc.

SAS Institute Inc., “The SAS System for Windows, Release 8.02, SAS System Help”, *CALL SOUND*, Cary, NC: SAS Institute Inc.

TRADEMARK CITATION

ACKNOWLEDGEMENTS

The author would like to express his appreciation to the following people for their assistance in this paper:
Wendy Cuthbert, AAI Development Services

CONTACT INFORMATION

Contact the author at:

David Fielding
AAI Development Services
Mississauga, Ontario, L5N 2W3
Work Phone: 905-816-9442
Fax: 905-816-9536
E-mail: david.fielding@aaidevelopment.com