

```
1 -----
2 --
3 --          GNAT RAVENSCAR for NXT          --
4 --          Copyright (C) 2010-2011, AdaCore --
5 --
6 -----
7
8 -- Based on the sound driver provided by the LeJOS project.
9
10 -- Expected usage is to declare a File object representing a wav file and
11 -- import it as follows:
12 --
13 --     Beep : NXT.Audio.Wav.File;
14 --     pragma Import (C, Beep, "beep_wav_start");
15 --
16 -- You can then play the file via procedure Play, where "My_Volume" is some
17 -- allowed volume value:
18 --
19 --     Play (Beep, My_Volume);
20 --
21 -- The wav file must be specified to the linker to satisfy the pragma Import.
22 -- The base name of the file must correspond to the first part of the name
23 -- used in the Import pragma. For the example above, the file name specified
24 -- to the linker would have a base name of "beep_wav". The file name extension
25 -- would not be ".wav" however, because the wav file must be converted to ELF
26 -- format and must have a specific symbol defined. To do the conversion you
27 -- can use the following in a makefile, or perform the indicated steps
28 -- manually. The conversion need only be done once. The resulting new file
29 -- will have an extension of ".owav" when using the steps below, but the
30 -- specific extension is not important as long as that same name is specified
31 -- to the linker.
32 --
33 -- arm-eabi-objcopy -I binary -O elf32-littlearm -B arm \
34 --   --redefine-sym _binary_hello_wav_start=hello_wav_start \
35 --   --redefine-sym _binary_hello_wav_end=hello_wav_end \
36 --   --redefine-sym _binary_hello_wav_size=hello_wav_size \
37 --   hello.wav hello.owav
38 --
39 -- Note how the symbols are altered. That is how the name string used in the
40 -- pragma Import is defined.
```

```
41
42 package NXT.Audio.Wav is
43
44   type File is limited private;
45   -- A "wav" file. Currently we support only 8-bit PCM data formats.
46
47   procedure Play (This : File; Volume : Allowed_Volume);
48
49   Invalid_Format : exception;
50
51 private
52
53   type File is limited null record;
54   -- The type must be actually limited (or tagged) in the full view because
55   -- it must be passed by reference for the sake of taking the address of
56   -- parameters of the type when passed to subprograms.
57   --
58   -- A wav file is a contiguous region of memory that contains subregions
59   -- referred to as "chunks". Each chunk is a descriptor that contains
60   -- specific data. The size of these data, and thus the chunk itself, vary
61   -- with the kind of chunk. Some kinds of chunk are mandatory, others are
62   -- optional. In addition, the required order of the chunks is only
63   -- partially defined. Therefore, we cannot easily define a single type that
64   -- statically represents all the possible content of a wav file. Hence we
65   -- declare a minimal chunk representation and use address arithmetic to
66   -- access the chunks within a file.
67
68   for File'Alignment use Unsigned_32'Alignment;
69   -- We map the first chunk in a file to the address of the file itself. The
70   -- alignment clause ensures the alignment of objects of type File will be
71   -- sufficient for the mapped "chunk" objects.
72
73 end NXT.Audio.Wav;
74
```