mp32dao documentation

Giuseppe Corbelli

Joe Steward

mp32dao documentation

by Giuseppe Corbelli and Joe Steward

Table of Contents

What is it?

mp32dao is a simple perl script designed to help you in the task of creating audio CDs from MP3 files. It will decode MP3 files to Wav and create a *TOC* file suitable for cdrdao.

Chapter 1. Authors

1.1. The beginning and nowadays

The creator of mp32dao is *Joe Steward*. He had the idea of track padding. It seems he's not maintaining mp32dao anymore, or at least I'm unable to contact him.

My name is *Giuseppe Corbelli* but I bet you saw it on the very first page of this document ;-). I made some hacking on the original program adding CD-Text support and internal mp3 decoding. I also added some more checks and rewrote all the stuff to improve readability and modularity. You should not think of finding a good programming example, though :-). You can contact me at <cowo@lugbs.linux.it>.

Chapter 2. Features

2.1. Why you’ll love mp32dao

Suppose you have downloaded a bunch of mp3 files. Suppose you want to burn them on a CD, without those annoying clicks between the tracks and with CD-Text support. This is why you’re going to use mp32dao.

Samples from the ending of a track are padded together with the beginning of the next to avoid leaving a CD sector half-filled with zeroes. You can avoid clicks between tracks this way.

By using ID3 tags such as ALBUM, AUTHOR, TITLE mp32dao will generate a toc-file with CD-Text info ready to use with cdrdao. Please take care that all MP3 files have the same AUTHOR and ALBUM ID3 fields. If don’t mp32dao will name the album "mp3 compilation" and author "VV.AA". Of course this does not apply if you are really creating a compilation. No ID3s mean no CD-Text support.

Chapter 3. Installation

3.1. Requirements

In order to be able to use mp32dao you must have, installed and working, on your system the following software:

Perl interpreter

I currently use:

```
[cowo@braveheart mp32dao]$ perl -V
Summary of my perl5 (revision 5.0 version 6 subver-
sion 0) configuration:
  Platform:
    osname=linux, osvers=2.2.18, archname=i686-linux
    uname='linux braveheart 2.2.18 #3
      tue feb 6 08:32:20 cet 2001 i686 unknown '
    config_args="
    hint=previous, useposix=true, d_sigaction=define
    usethreads=undef use5005threads=undef
      useithreads=undef usemultiplicity=undef
      useperlio=undef d_sfio=undef uselargefiles=undef
      use64bitint=undef use64bitall=undef
      uselongdouble=undef usesocks=undef
  Compiler:
    cc='cc', optimize='-O3 -fomit-frame-pointer
      -mpentiumpro -march=pentiumpro -malign-functions=4
      -funroll-loops -fexpensive-optimizations -malign-double
      -fschedule-insns2 -mwide-multiply',
      gccversion=2.95.3 19991030 (prerelease)
    cppflags='-fno-strict-aliasing -I/usr/local/include'
    ccflags ='-fno-strict-aliasing -I/usr/local/include'
    stdchar='char', d_stdstdio=define, usevfork=false
    intsize=4, longsize=4, ptrsize=4, doublesize=8
```

```
      d_longlong=define, longlongsize=8, d_longdbl=define,
        longdblsize=12
      ivtype='long', ivsize=4, nvtype='double', nvsize=8,
        Off_t='off_t', lseeksize=4
      alignbytes=4, usemymalloc=n, prototype=define
    Linker and Libraries:
      ld='cc', ldflags =' -L/usr/local/lib'
      libpth=/usr/local/lib /lib /usr/lib
      libs=-lnsl -lgdbm -lpthread -ldl -lm -lc -lcrypt
      libc=/lib/libc-2.1.92.so, so=so, useshrplib=true,
        libperl=libperl.so
    Dynamic Linking:
      dlsrc=dl_dlopen.xs, dlext=so, d_dlsymun=undef,
        ccdlflags='-rdynamic -Wl,-rpath,
        /usr/lib/perl5/5.6.0/i686-linux/CORE'
      cccdlflags='-fpic', lddlflags='-shared -L/usr/local/lib'


  Characteristics of this binary (from libperl):
    Compile-time options:
    Built under linux
    Compiled at Feb  7 2001 14:50:21
    @INC:
      /usr/lib/perl5/5.6.0/i686-linux
      /usr/lib/perl5/5.6.0
      /usr/lib/perl5/site_perl/5.6.0/i686-linux
      /usr/lib/perl5/site_perl/5.6.0
      /usr/lib/perl5/site_perl
      .
```

and it works for me. Please note I compiled Perl myself, disabling thread support. You may get into trouble if you want to use internal mp3 decoding coupled with a threaded Perl distribution. This will be made clear later.

MP3::Info Perl Module

It is used to access MP3 informations such as playing length, ID3 tags and more.

Audio::Wav Perl Module

It is used to access Wav file information such as length.

Audio::Tools::Time Perl Module

Almost the same as above.

mp3handler Perl Module

Contains the object I use for MP3 handling. You may find it useful yourself. You can find it along with mp32dao. If you plan to install mp32dao system-wide you must put it somewhere in the perl search path. Or leave it in the same directory where you unpacked mp32dao.

mp3dec Perl Module

This module will be used to handle mp3 decoding internally, using libmpeg3 and libsndfile. It is not absolutely necessary; an external decoding program can be used such as lame or madplay. It will be used only if available. You can create this module yourself by using SWIG (http://www.swig.org), following these steps:

1. compile `mp3dec.c`, for example using

   ```
   gcc -fPIC -D_REENTRANT -c mp3dec.c
   ```

   You will obtain `mp3dec.o`. This is a stupid console program which exports a function used to decode mp3 files to wav files.

2. generate perl wrapper using SWIG:

   ```
   swig -perl5 mp3dec.i
   ```

   You will obtain files `mp3dec_wrap.c` and `mp3dec.pm`. The first will be used by perl to access function exported by mp3dec.c, the second is the one to include in perl code.

   Note: If you are using threaded perl SWIG won't work! (at least for me). Please use a no-threads perl build.

3. compile SWIG-generated wrapper:

```
gcc -fPIC -c -Dbool=char -I /usr/lib/perl5/5.6.0/i686-
linux mp3dec_wrap.c
```

Of course the include option should match the right one for your system. On Redhat Linux systems it is usually `/usr/lib/perl5/5.6.0/i386-linux` on Intel machines.

4. create the shared library that will be loaded by Dynaloader:

```
gcc -shared mp3dec_wrap.o mp3dec.o -o mp3dec.so -lm -
lpthread -lmpeg3 -lsndfile
```

Ok, as you can see there are some external libraries needed; libmpeg3 and libsndfile which you'll find on freshmeat.net. libm and libpthread are standard on glibc2 systems, I don't have access to others, sorry. Then you must copy the resulting `mp3dec.so` somewhere in the perl search path.

3.2. Compiling and installing

If you have SWIG, libpthread, libmpeg3 and libsndfile already installed you may use internal mp3 decoding module. Otherwise you must have an external mp3 decoding software available. There are no particular differences, but internal decoding gives a better feeling :-)).

3.2.1. Internal Decoding

Edit the supplied `Makefile` to fit your system's configuration. Then type `make` and `make install`.

3.2.2. External Decoding

You must have an external command-line mp3 decoder installed. Then edit `mp32dao.pl`, find the line `my $external_mp3decoder = 'lame -decode`

`-mp3input';` and change the programname and options to suit your needs. Please note that the external program must take a command-line like this `programname [OPTIONS] input.mp3 output.wav` to work with mp32dao. You can leave mp32dao anywhere you like but take care that `mp3handler.pm` is in its same directory or somewhere in the perl search path.

Chapter 4. Usage

4.1. Command line options

The usage is fairly straightforward:

1. put all and only the mp3files you want on the CD in a single directory.
2. change to this directory
3. launch mp32dao.pl [tocfile]. If you specify a tocfile it will be used to write the file .toc used by cdrdao. If you don't `cd.toc` will be used. Other arguments are ignored.

The script should work with all filenames but you'll live happier if you don't name your files like this:

```
Iced % Earth - Burning ^ Times - [01].mp3
```

Avoid stupid naming schemes. Please note the program will compose the CD using the files in the same order as their name suggests. I *strongly* advise you to put the tracknumber at the beginning of the filename. Example:

```
01-The_Ghosts_Of_Christmas_Eve.mp3
02-Boughs_Of_Holly.mp3
03-The_World_That_She_Sees.mp3
04-Midnight_Christmas_Eve.mp3
05-The_March_Of_The_Kings-Hark_The_Herald_Angel.mp3
```

Chapter 5. Internals

5.1. Program structure

Still to be written. But take a look at the code, it should be clear enough.

Chapter 6. The future

6.1. Next release plans

On-the-fly decoding

No more huge wav files. It should be possible to do the decoding on the fly using a FIFO.

Packaging, documentation

Decent packaging with makemake support and readable documentation.

Ogg support

Support for ogg files is on the way, don't worry :-).

Better internal decoding module

Ah, you saw how it sucks? :-) It's only a quick hack over libmpeg3. It should be much better, but sometimes it works :-)

6.2. Sooner or later

I'd like to couple the internal decoding module with a threaded perl. It won't probably speed up things much but will give a *definite* good feeling :-). Up till now SWIG and threaded perl don't work together. Seems we have to wait.

Chapter 7. Troubleshooting

7.1. segfaults and nasty bugs here

There’s no point in writing such a section since I develop high quality software :-)).

Chapter 8. Developing

8.1. Wanna help me?

I’m working on mp32dao alone. I always need help and knowledge. Please drop me a mail at <cowo@lugbs.linux.it>