

# Managing Digital Music Collections with Perl

Paul Mison  
YAPC::Europe  
24 July 2003

Managing Digital Music Metadata with Perl

# ID3: Why and How

## Digital Music Basics

- What is an MP3?
  - It's a digital music file
- Why metadata?
  - Filenames aren't rich or portable
- What format?
  - ID3 is the standard (even for Ogg Vorbis, kinda)

## ID3: Really useful?

- Audio applications
  - Players, like mpg123, WinAmp
  - Catalogues, like iTunes
- Portable devices (iPod)
- Fixed devices (SliMP3)
- Your own sense of order

## ID3: nest of standards

- ID3v1: first attempt
- Simple data at the end of the file
- Limited though
  - 30 character limit
  - Only seven fields



## Second system syndrome

- ID3v2 - rewriting it
- Expressive
- Complicated
  - Many, many variable length fields
  - Beginning of file (except ID3v2.4)
  - Hence hard to parse

## Which to use?

- Modern apps read both
  - tend to write ID3v2
- v1 too limited; v2's richness is useful
- Only problem- hard to write
- ... even with Perl, sadly

## Perl and ID3

- Three obvious module choices
  - MP3::Info
  - MP3::Tag
  - MP3::ID3Lib

## MP3::Info

- Functional / hybrid interface
- Pure Perl
- Can read and write v1
- Can only read v2
  - (but see `MP3::Info::set_mp3v2tag`)

## MP3::Tag and MP3::ID3Lib

- OO interfaces
- Read and write both v1 and v2
- Need some knowledge of the spec
  - Tag is pure Perl
  - ID3Lib requires that C library

## Bridging the gap

- MP3::Info has the easiest interface
- MP3::Tag is the most capable
- Merge the two: MP3::Info::set\_mp3v2tag
  - Same interface as MP3::Info
  - Uses MP3::Tag to write out

Managing Digital Music Metadata with Perl

# Correcting Bad Metadata

## Using m3u to populate ID3

- m3u are MP3 playlists
- Two modules:
  - MP3::M3U
  - MP3::M3U::Parser
- Fairly simple script

## Rename from tags

- Sometimes filenames are messy
- Rename them from ID3 tags
- Example script
- You can also go the other way...

## Dreaded 'Various Artists'

- Compilations have 'artist / title'
- Applications sometimes broken
- Again, simple script
- Have to be careful
  - Script can't determine order

Managing Digital Music Metadata with Perl

# Using web services

## A Digression

- Previous slides worked with existing data
- Sometimes you don't have that luxury
  - (insert Tom's joke here)
- Need somewhere to look up data
- Ideally, automated

## CD Metadata History

- In the beginning was the CDDB
- Submit CD information
- In exchange, save typing
- Gracenote 'closed' the DB
- GPL data to that point still free
- Used by two open source projects

## The two OS projects

- FreeDB
  - Very minimal CDDB expansion
  - Web search but no more
- MusicBrainz
  - Much more ambitious
  - RDF and web services

## Webservices::FreeDB

- As noted, only a web service
- Module uses LWP::Simple, regexes
- Looks a little fragile
- Might be good for you

## MusicBrainz::Client

- Uses proper web services
- Relies on MusicBrainz C library
- ... and it shows, too
- Can extract RDF, if you want

## Using MusicBrainz

- Set up object and query
- Get data from query
- Loop over data
- Perhaps apply it to the source MP3

# Managing Digital Music Metadata with Perl

## Some sample code

```
sub handle_album_track_list {
    my $result;

    for (my $i = 1;; $i++) {
        $mb->select(MBS_Rewind);
        if (!$mb->select1(MBS_SelectLookupResult, $i)) {
            last;
        }
        my $relevance = $mb->get_result_int(MBE_LookupGetRelevance);

        # get track info
        $mb->select(MBS_SelectLookupResultTrack);
        my $track    = $mb->get_result_data(MBE_TrackGetTrackName);
        my $length  = $mb->get_result_data(MBE_TrackGetTrackDuration);
        my $artist  = $mb->get_result_data(MBE_TrackGetArtistName);
        $mb->select(MBS_Back);

        # get album info
        $mb->select(MBS_SelectLookupResultAlbum);
        my $album    = $mb->get_result_data(MBE_AlbumGetAlbumName);
        my $ids      = $mb->get_result_int(MBE_AlbumGetNumCdindexIds);
        my $trackct  = $mb->get_result_int(MBE_AlbumGetNumTracks);

        $mb->select(MBS_Back);

        $result->[$i] = { relevance => $relevance,
                        track     => $track,
                        album     => $album,
                        artist    => $artist,
                        total     => $trackct,
                        time      => $length,
                        };
    }
}
```

## An example

- So, here's a script
- It's a bit scrappy inside
- Works, though
  - Mainly
- Almost magical

## Further topics

- Controlling iTunes with Perl
  - Mac::Glue or Mac::iTunes
  - Parsing library with Mac::PropertyList
- Similarly for XMMS, mpg123
  - Both present challenges, but can be done
  - Winamp too, but I don't do Windows

## Further topics

- MusicBrainz without the API
- Submitting via FreeDB module
- Apache::MP3
- MP3 as extension
  - File::Find::Rule::MP3Info, Template::MP3
- ... and more

## To sum up

- Perl can edit ID3 tags
  - but some more work would be nice
- Fairly simple to tidy or make ID3 tags
- Web services can be used to identify MP3s
  - again, a little complex still

# Managing Digital Music Metadata with Perl

**Thank you**

**Any questions?**