



AGH University of Kraków

Project Documentation

FLAC file audio player

For the course

Object-Oriented Programming

EiT, III year

Michał Charaszkiewicz

Friday 15:00

instructor: Jakub Zimnol

09.01.2024

1. Project description

The app developed during this project allows for real-time decoding and playback of audio files in the FLAC format. The application opens a file given to it as an argument when running it in the terminal and plays back the audio encoded in it, while displaying the artist's name, track title and album name. It will play the audio continuously, until it reaches the end of the file

2. Classes

a. Flac class

This class handles decoding all the necessary metadata, as well as the audio samples themselves. It only provides 2 public methods (apart from getters):

- initialize() for verifying the provided file is a valid FLAC file and extracting the data necessary for playback, as well as the metadata containing track information (title, album etc.)

- decode_frame(), which decodes 1 full frame of audio data, with the samples stored in an audio buffer, which is a member of the class and can be accessed through a getter for playback in the main function

This class contains mostly private methods, that should not be called by the user as they should only be called at specific points in the file. Because of that, those methods don't have doxygen comments.

b. Bit_reader class

This template class provides functionality for reading non standard numbers of bits (not multiples of 8) from a provided stream (in the case of this player a std::ifstream) both as signed and unsigned integers. It also provides functionality to align back to a byte (useful when the data is zero-padded to a byte boundary) and an eos() method to check whether the stream has ended without reading it and potentially crashing the program.

3. Playback functionality

For playback I decided to utilize the ALSA API (Advanced Linux Sound Architecture), which is a low level audio library for Linux. For this reason, to compile the project, the ALSA development library needs to be installed, which on Debian/Ubuntu can be done like this:

```
sudo apt-get install libasound2-dev
```

This only needs to be done for the purpose of compilation, as ALSA itself is part of the Linux kernel. If a compiled executable were to be provided, no additional libraries need to be installed to run the application.

If the application is to be run in a WSL environment, [these](#) instructions should be followed, as ALSA works directly with hardware, and WSL, does not have a sound card. For playback, the audio must be routed through pulseaudio into windows to play it.

4. Building

The project utilizes CMake for the build system. The executable can be built by running the following commands in the root directory of the project.

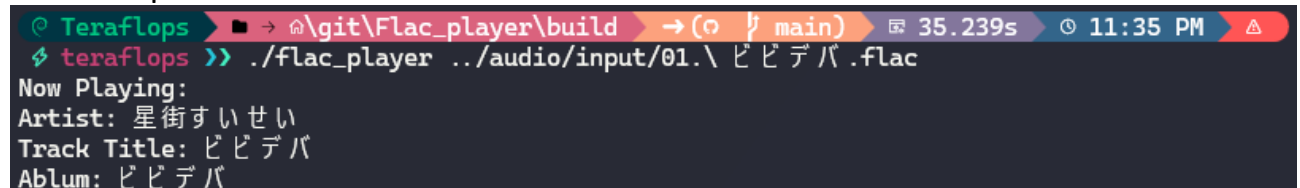
```
mkdir build
cd build
cmake ..
cmake --build . --config Release
```

5. Usage example

The compiled program can be used like this:

`./flac_player <relative_path_to_file>`

For example:

A terminal window with a dark background and a multi-colored header bar. The header bar shows the user 'Teraflops', the current directory path '\git\Flac_player\build', the branch 'main', a timer at '35.239s', and the time '11:35 PM'. The terminal text shows the command './flac_player ../audio/input/01.\ ビビデバ.flac' being executed. Below the command, it says 'Now Playing:' followed by 'Artist: 星街すいせい', 'Track Title: ビビデバ', and 'Album: ビビデバ'.

```
@ Teraflops > \git\Flac_player\build -> (main) 35.239s 11:35 PM
teraflows >> ./flac_player ../audio/input/01.\ ビビデバ.flac
Now Playing:
Artist: 星街すいせい
Track Title: ビビデバ
Album: ビビデバ
```