CS 3005: Programming in C++

WAV File Creator

Introduction

A WAV file is a binary file that contains audio data. The quality of the audio data is determined by the sample rate and the bits per sample. The duration of the sound is determined by the number of samples. The sound is controlled by the form of the audio data.

Assignment

In this assignment, you will make a program that allows the user to configure a WAV file, including the two audio tracks stored in it to make a stereo sound file.

The program should create a WAV file from two audio tracks designed by the user. The user will choose the "Samples/Second" (the sample rate), the "Seconds" (the duration of the audio tracks), and the "Bits/Sample[8,16,24,32]" the number of bits per sample for the WAV file. The user will also be able to select the form of the audio data for the left and right channels separately. Finally, the user will allowed to choose the name of the WAV file to be stored on the drive. The audio track form will be configured using the functions built in previous assignments.

An interaction with the program may look like this:

```
$ ./program-wav-file-creator/wav_file_creator
Samples/Second: 44100
Seconds: 1.75
Bits/Sample[8,16,24,32]: 16
Left Channel
Fill style: sine
Frequency: 440
Right Channel
Fill style: sawtooth
Frequency: 880
WAV filename: sample.wav
$ ls -l sample.wav
-rw-rw-r-- 1 cgl cgl 308744 Sep 19 14:51 sample.wav
```

You can download and listen to sample.wav.

Programming Requirements

Update `library-application/ApplicationData.{h,cpp}`

`ApplicationData` Class

Data Members:

- `WAVFile` object for the current application. Should be created with the sample rate of 1 and bits per sample of 8 in `ApplicationData`'s constructor.
- `std::vector<AudioTrack>` object for the current application. This will be used as an assembly area for building the channels for the WAV file. Should be created with size 0 in `ApplicationData`'s constructor.

`public` Methods:

- `WAVFile& getWAVFile();` Returns a reference to the wav file member.
- `const WAVFile& getWAVFile() const;` Returns a const reference to the wav file member.
- `std::vector<AudioTrack>& getChannels();` Returns a reference to the channels member.
- `const std::vector<AudioTrack>& getChannels() const;` Returns a const reference to the channels member.

Create `library-commands/wav_file_creator_aux.{h,cpp}`

Functions:

- `void configure_audio_track_and_wav_file(ApplicationData& app_data);` Prompts the user for "Samples/Second: ", "Seconds: ", and "Bits/Sample[8,16,24,32]: ", then sets the values in the `ApplicationData` object's `WAVFile` and `AudioTrack` members. Pay attention to the types of the variables when prompting the user. You should be using `ApplicationData`'s `getInteger()` and other methods to fetch values from the user.
- `void fill_channels(ApplicationData& app_data);` Size the channels vector from the `ApplicationData` object to have 2 channels. 0 is the left channel, and 1 is the right channel. For each channel, display a message to the user indicating the channel ("Left Channnel" or "Right Channel"), then use `fill_audio_track` to allow the user to configure the audio track for that channel. Finally assign into the correct position in the channels vector from the `ApplicationData` object's `AudioTrack` member. This function can assume the `ApplicationData` object's `AudioTrack` object has its meta data already configured.
- `void save_wav_file(ApplicationData& app_data);` Prompt the user for "WAV filename: ", then write the WAV file that is stored in the `app_data` to disk. Assumes the WAV file and channels vector have already been configured.
- `int wav_file_creator(ApplicationData& app_data);` Create a vector of `AudioTrack` objects. Use `configure_audio_track_and wav_file()` to get meta data from the user. If the user's configuration choices are valid (as determined by the `AudioTrack`'s size being greater than 0), then `fill_channels` and `save_wav_file`. Otherwise, display the error message "Positive values expected for samples per second and seconds.". Finally, return the size of the `AudioTrack` object.

Update `library-commands/Makefile`

Add `wav_file_creator_aux.{h,cpp}` in the appropriate places to add them to the library and install the header file.

Create `program-wav-file-creator/wav_file_creator.cpp`

Functions:

- `int main();` Entry point to the wav file creator program. Should create an `ApplicationData` and pass it to the `wav_file_creator` function found in `wav_file_creator_aux` and return the result of that function call.

Create `program-wav-file-creator/Makefile`

This file must contain rules such that any of the following commands will build the `wav_file_creator` program:

- `make`
- `make all`
- `make wav_file_creator`

Create `program-wav-file-creator/.gitignore`

The file `program-wav-file-creator/.gitignore` needs to store one line of text:

```
wav_file_creator
```

This will prevent the executable program from being committed to the repository. It is a *derived file*.

Update `Makefile`

- Update the project-level Makefile so that `make` and `make all` in the project directory will call `make` in the `program-wav-file-creator` directory.
- If necessary, make sure the order of make commands is correct to build prerequisite libraries in the correct order.

Additional Documentation

- None

Grading Instructions

To receive credit for this assignment:

- your code must be pushed to your repository for this class on GitHub

- all unit tests must pass
- all acceptance tests must pass
- all programs must build, run, and execute as described in the assignment descriptions.

Extra Challenges (Not Required)

- None