Spike: 8
Title: Sound Board

Author: Ryan Chessum, 102564760

Goals / deliverables:

- Code, see /12 – Spike – Sound Board/Sound Board/
- Spike Report

Technologies, Tools, and Resources used:

List of information needed by someone trying to reproduce this work

- Visual Studio 2019
- C plus plus reference (https://www.cplusplus.com/reference/)
- Royalty Free Sound effects and music:
  - https://pixabay.com/sound-effects/jazz-loop-7163/
  - https://bigsoundbank.com/detail-1758-star-wars-blaster-2.html
  - https://bigsoundbank.com/detail-1139-firecracker-with-wick-3.html
  - https://bigsoundbank.com/detail-1860-cap-cider-3.html
- SDL2 (https://www.libsdl.org/)
- SDL2 Mixer (https://www.libsdl.org/projects/SDL_mixer/)

Tasks undertaken:

- Download and install Visual Studio
- Create a new C++ project
- Download SDL development libraries
- Link SDL Library to project
- Download SDL2 Mixer development libraries
- Link SDL Mixer to project
- Find music and sound effects and add them to the project files
- Add code to load in the sounds
- Add code to play each sound with a button press
- Play the music before the gameloop
- Add code to play or pause the music with a keypress
- Make sure memory is freed at the end of the program

What we found out:

To use sound effects and music with SDL2, an add on library called SDL mixer is need. SDL mixer provides objects and structs for music and sound as well as a mixer to play them for us.

To use SDL mixer you need to download and link the development libraries as well as the development libraries for SDL2.

Once both libraries are set up, we can use SDL mixer to create a simple sound board program.

I found some good royalty free music and sound effects to use for the program linked in the resources. They should be added somewhere in the project files.

SDL mixer contains two datatypes for sound. Mix_Music is for music and Mix_Chunk is for sound effects (think about it like a chunk of sound).

```
//The music that will be played
Mix_Music* jazz_music = nullptr;

//The sound effects that will be used
Mix_Chunk* gun_SE = nullptr;
Mix_Chunk* explode_SE = nullptr;
Mix_Chunk* pop_SE = nullptr;
```

I added some pointers for sound effects and a pointer for music for the sounds I want to use.

```
int main(int argc, char* args[])
{
    if (SDL_Init(SDL_INIT_VIDEO | SDL_INIT_AUDIO) < 0)
    {
        printf("SDL could not initialize! SDL Error: %s\n", SDL_GetError());
        success = false;
    }
    else
    {
        int rate = 44100;
        Uint16 format = MIX_DEFAULT_FORMAT;
        int channleNo = 2;
        int buffers = 2048;
        if (Mix_OpenAudio(rate, format, channleNo, buffers) < 0)
        {
            printf("SDL_mixer could not initialize! SDL_mixer Error: %s\n", Mix_GetError());
            success = false;
        }

        //Create window
        window = SDL_CreateWindow("SDL DEMO", SDL_WINDOWPOS_UNDEFINED, SDL_WINDOWPOS_UNDEFINED, WIDTH, HEIGHT, SDL_WINDOW_SHOWN);
        if (window == nullptr)
        {
            printf("Window could not be created! SDL_Error: %s\n", SDL_GetError());
            success = false;
        }
        else
        {
            screen = SDL_GetWindowSurface(window);

            //Fill the surface green
            SDL_FillRect(screen, nullptr, SDL_MapRGB(screen->format, 255, 255, 255));

            //Update the surface
            SDL_UpdateWindowSurface(window);
        }

        success = loadSounds();
    }
```

Next, I added code in main to initialise everything before the game loop. Some of it is similar or the same as in Task 15.

To start using audio we need to call Mix_OpenAudio(). It opens an audio mixer with the frequency, number of channels, format and sound effect size we specify. Two channels means we are using stereo audio.

If it fails to initialise it will set a Boolean variable to false and print an error message to the console.

After we have that we can load the sounds in. I made a separate function for this. The Mix_LoadMUS() and Mix_LoadWAV() functions take in the directory of the file you want to load in and create a music or chunk instance of that sound in memory. Your files need to be of .WAV format for this to work. The functions return a pointer, so we can set our variables to the pointer returned by each function. If it fails to load the music it returns an error again and sets the success variable to false.

```
bool loadSounds()
{

    //Load music
    jazz_music = Mix_LoadMUS("jazz-loop-7163.wav");
    if (jazz_music == nullptr)
    {
        printf("Failed to load beat music! SDL_mixer Error: %s\n", Mix_GetError());
        success = false;
    }

    //Load sound effects
    gun_SE = Mix_LoadWAV("LASRGun_Star wars blaster 2 (ID 1758)_BSB.wav");
    if (gun_SE == nullptr)
    {
        printf("Failed to load scratch sound effect! SDL_mixer Error: %s\n", Mix_GetError());
        success = false;
    }

    explode_SE = Mix_LoadWAV("FRWKRec_Firecracker with wick 3 (ID 1139)_BSB.wav");
    if (explode_SE == nullptr)
    {
        printf("Failed to load high sound effect! SDL_mixer Error: %s\n", Mix_GetError());
        success = false;
    }

    pop_SE = Mix_LoadWAV("FOODGware_Cap cider 3 (ID 1860)_BSB.wav");
    if (pop_SE == nullptr)
    {
        printf("Failed to load medium sound effect! SDL_mixer Error: %s\n", Mix_GetError());
        success = false;
    }
    return success;
}
```

Sine sounds require a lot more space compared to data we normally work with; it is allocated to the heap meaning we must make sure the memory is

freed at the end. I made another function to handle the freeing of memory space.

```
void close()
{
    //Free the sound effects
    Mix_FreeChunk(gun_SE);
    Mix_FreeChunk(explode_SE);
    Mix_FreeChunk(pop_SE);
    gun_SE = nullptr;
    explode_SE = nullptr;
    pop_SE = nullptr;

    //Free the music
    Mix_FreeMusic(jazz_music);
    jazz_music = nullptr;

    //Destroy window
    SDL_FreeSurface(screen);
    SDL_DestroyWindow(window);
    window = nullptr;
    screen = nullptr;

    //Quit SDL subsystems
    Mix_Quit();
    SDL_Quit();
}
```

The function is called after the game loop. There are special functions that handle the freeing of the sounds for us, we just need to make sure they are called when we are finished with them.

```
while (running && success)
{
    while (SDL_PollEvent(&e) != 0)
```

If everything loaded correctly, success should be true meaning the program can step into the game loop.

```
case SDLK_1:
    // Play Sound
    Mix_PlayChannel(-1, gun_SE, 0);
    break;
case SDLK_2:
    // Play Sound
    Mix_PlayChannel(-1, explode_SE, 0);
    break;
case SDLK_3:
    // Play Sound
    Mix_PlayChannel(-1, pop_SE, 0);
    break;
}
```

In the event handler in the game loop, when 1, 2 or 3 are pressed it will play one of the three sound effects. We can call each one with Mix_PlayChannel(). The first parameter is the channel we want to play the audio on. If we pass in -1 it will look for the first free channel available. This will let the sounds overlap with one another. The second parameter is a pointer to the sound we want to play. The last one is the number of times we want the sound to loop. We only want it to play once so we set it to 0. If you were to set it to -1 it would loop indefinitely.

```
if (success && Mix_PlayingMusic() == 0)
{
    //Play the music
    Mix_PlayMusic(jazz_music, -1);
}
```

The music works slightly differently. We can play music using Mix_PlayMusic(). We pass in the pointer to the music and the number of times we want it to loop. I set it to -1 to play on repeat. Then in the event handler in the game loop we can pause and play the music when 0 is pressed.

```
while (SDL_PollEvent(&e) != 0)
{
    if (e.type == SDL_QUIT)
    {
        running = false;
    }

    //Get keydown
    else if (e.type == SDL_KEYDOWN)
    {
        switch (e.key.keysym.sym)
        {
        case SDLK_0:
            //If the music is paused
            if (Mix_PausedMusic() == 1)
            {
                //Resume the music
                Mix_ResumeMusic();
            }
            //If the music is playing
            else
            {
                //Pause the music
                Mix_PauseMusic();
            }

            break;
        case SDLK_1:
```

Music can be played or paused with Mix_PauseMusic() and Mix_ResumeMusic(). We check the status of the music With Mix_PausedMusic(). If it is paused it will resume the music for us. Otherwise it will pause it.

Now everything should be working perfectly, and you should have a working soundboard.