Sound Programming

제8장 Ogg Vorbis Playback in OpenAL

Overview of Ogg Vorbis

- **An open-source sound codec created by Xiphophorus**
  - www.xiph.org
  - popular on Linux due to its open-source heritage
  - an alternative to MP3
  - Wav vs. Ogg Vorbis
- **Ogg Vorbis allows you to decode the format yourself without having to pay a fee.**
- **Ogg Vorbis *vs.* MP3 *vs.* WMA ?**
  - Which one do you like better ?

Ogg Vorbis의 구조

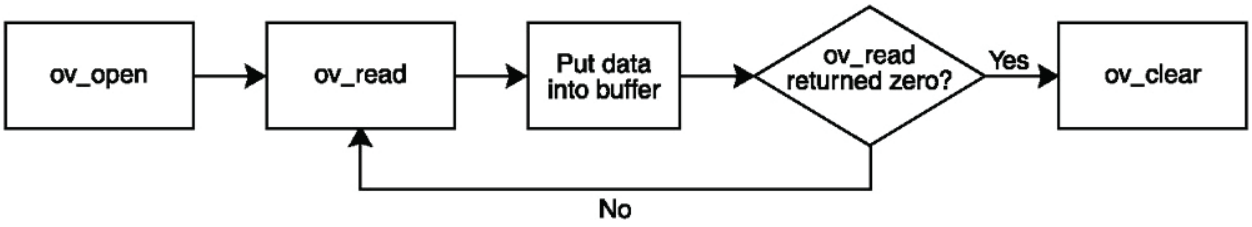
- **Ogg Vorbis API**는 크게 **4** 개의 라이브러리로 구성
- **Ogg library**
  - Ogg 파일을 다루기 위한 general-purpose routine들을 지원
  - Ogg 라이브러리는 Vorbis를 알지 못함.
  - Ogg 라이브러리는 실제 다루는 파일이 사운드인지 동영상인지, 이미지 인지도 구분하지 않음
  - 멀티미디어 데이터 처리를 위한 **"base framework"**를 제공할 뿐임.
- **Vorbis library**
  - Vorbisenc를 이용하여 Vorbis 방식으로 압축된 사운드 스트림을 디코딩할 때 필요한 함수들을 제공한다.

Ogg Vorbis의 구조 (계속)

- **Vorbisenc library**
  - Vorbis 파일로의 인코딩을 지원하는 라이브러리
  - Vorbis 인코더 함수 일체를 지원한다.
  - CD를 Ogg Vorgis 파일로 만든다거나, Wav 파일을 Ogg Vorbis 파일로 만드는 등의 일을 할 때에는 꼭 필요한 라이브러리 임.
- **Vorbisfile library**
  - *Ogg Vorbis 파일을 쉽게 다룰 수 있게 해주는 편의 라이브러리*
  - Ogg와 Vorbis 라이브러리들을 감싸고 있으며, 사용하기 편리한 인터페이스를 제공한다.
  - 특히, Vorbis 파일을 디코딩할 때 편리한 기능을 제공

Using the Vorbisfile API

- **Ogg Vorbis API**를 감싸 매우 사용하기 편하게 만들어 놓은 라이브러리
  - Ogg Vorbis 파일을 읽고 압축을 해제하여, 이를 PCM 스트림으로 변환하는 과정을 아주 편리한 몇 개의 함수만으로 지원함.
  - Vorbisfile API를 사용하는 flowchart



  - 이번 장에서 우리는 Ogg Vorbis API 중 Vorbisfile API 만을 사용할 것임!

Setup/Teardown API

- **ov_open_callbacks()**
  - 주어진 file handle로부터 Ogg Vorbis bitstream을 초기화하고,
  - custom file/bitstream manipulation 루틴들을 초기화한다.

```
int ov_open_callbacks(
    void *datasource,          // 이미 오픈된 .ogg 파일의 FILE *를 넘기면 됨
    OggVorbis_File *vf,        // 정보가 채워질 OggVorbis_File 구조체를 가리키는 포인터
    char *initial,             // Typically set to NULL
    long ibytes,               // Typically set to 0
    ov_callbacks callbacks     // A completed ov_callbacks struct which indicates desired
                               // custom file manipulation routines
);
```

  - 리턴 값
    - 0 for success
    - less than zero for failure

Setup/Teardown API (계속)

- ov_open_callbacks() (계속)
  - ov_callbacks 구조체

```
typedef struct {
    size_t (*read_func)  (void *ptr, size_t size, size_t nmemb, void *datasource);
    int    (*seek_func)  (void *datasource, ogg_int64_t offset, int whence);
    int    (*close_func) (void *datasource);
    long   (*tell_func)  (void *datasource);
} ov_callbacks;
```

- ov_clear()
  - bitstream과 .ogg 파일을 닫고, cleans up loose ends.
  - Must be called when finished with the bitstream

```
int ov_clear(OggVorbis_File *vf);
```


ogg 파일 정보 얻기

- ov_info()
  - 주어진 ogg bitstream의 정보를 담은 vorbis_info 구조체 포인터를 리턴한다.

```
vorbis_info *ov_info(
    OggVorbis_File *vf,      // OggVorbis_File 구조체를 가리키는 포인터:
                             // ov_open_callbacks() 으로부터 넘겨 받은 것이어야 함.
    int link                 // 대부분의 경우 -1
);
```

  - 리턴 값: 정보가 채워진 vorbis_info 구조체를 가리키는 포인터
    - vorbis_info 구조체: vorbis/codec.h 파일에 정의되어 있음
    - 파일 버전, 채널 수, sample rate, Bitrate(CBR인 경우) 등의 정보를 담고 있음.

ogg 파일 정보 얻기(계속)

- **ov_pcm_total()**
  - 주어진 ogg bitstream에 있는 총 PCM 샘플링 횟수를 리턴한다.
  - 디코딩 했을 때 차지할 바이트 수 계산에 활용할 수 있음.

```
ogg_int64_t ov_pcm_total(
    OggVorbis_File *vf,
    int link                    // 대부분의 경우 -1
);
```


Decoding

- **ov_read()**

```
long ov_read(
    OggVorbis_File *vf, // OggVorbis_File 구조체를 가리키는 포인터
    char *buffer,       // 압축 해제된 PCM 데이터가 저장될 버퍼 포인터
    int length,         // buffer로 읽어 들일 바이트 수. 버퍼의 크기와 같은 크기
                        // 이어야 함. 대개 4096이면 적당.
    int bigendianp,     // Specifies big or little endian byte packing.
                        // 0 for little endian, 1 for big endian. 보통은 0.
    int word,           // word 크기를 지정한다. 1 for 8-bit samples,
                        // or 2 for 16-bit samples. 보통은 2.
    int sgned,          // Signed or unsigned data.
                        // 0 for unsigned, 1 for signed. 보통은 1.
    int *bitstream);    // A pointer to the number of the current logical bitstream.
```

  - 응용 프로그램에서 ogg 파일을 디코딩할 때 사용되는 주 함수.
  - 압축 해제된 PCM 오디오 데이터를 지정된 길이만큼 리턴한다.
    - endianness, signedness, 그리고 word 크기도 고려함.
  - 인코딩된 오디오가 multichannel이면, 각 채널들이 출력 버퍼에 인터리빙되어 채워진다.

Decoding (계속)

- ov_read()(계속)
  - Return Values
    - OV_HOLE
      - 파일 안에 빈 곳이 있어 디코딩에 실패함 (사유: 파일 페이지 사이에 쓰레기 데이터가 있음, recapture가 바로 따라오는 바람에 동기화 정보가 사라짐, 또는 파일 페이지 자체가 깨짐)
    - OV_EBADLINK
      - indicates that an invalid stream section was supplied to libvorbisfile, or the requested link is corrupt.
    - 0 indicates EOF
    - *n* indicates actual number of bytes read
      - ov_read()는 일단 호출되면 vorbis packet 단위로 디코딩 하기 때문에,
      - 이 리턴 값이 인자에 지정한 length 값보다 작을 수 있음.

.ogg 파일 재생 과정

- 단계 1: .ogg → Wav 스트림 변환 과정
  - Ogg Vorbis 파일을 open하고 decoding 한다.
  - 디코딩된 Ogg Vorbis 스트림은 압축되어 있지 않은 PCM stream 이라고 할 수 있다.
- 단계 2: 이 PCM stream을 OpenAL Buffer에 담는다.
- 단계 3: OpenAL Buffer와 Source를 연결하여 재생

8장 예제(PlayOggVorbis) main()

```
#include "Framework.h"
#include "Vorbis\vorbisfile.h"
#include "alLoadOggVorbis.h"

#define   SERVICE_UPDATE_PERIOD         250
#define   TEST_OGGVORBIS_FILE           "아이유-06-삼촌 (Feat. 이적).ogg"

int main()
{
    ALuint uiBuffer;
    ALuint uiSource;
    Alint  iState;

    // Initialize Framework
    ALFWInit();
    ALFWprintf("PlayOggVorbis Test Application\n");

    // Initialize OggVorbis libary
    InitVorbisFile();
    if (!g_bVorbisInit) {
        ALFWprintf("Failed to find OggVorbis DLLs (vorbisfile.dll,ogg.dll,or vorbis.dll)\n");
        ALFWShutdown();
        return -1;
    }

    // Initialise OpenAL
    if (!ALFWInitOpenAL()) {
        ALFWprintf("Failed to initialize OpenAL\n");
        ALFWShutdown();
        return -1;
    }
    ……
```


8장 예제(PlayOggVorbis) main() (계속)

```
    ……
    // Open the OggVorbis file
    FILE *pOggVorbisFile = fopen(ALFWaddMediaPath(TEST_OGGVORBIS_FILE), "rb");
    if (!pOggVorbisFile) {
        ALFWprintf("Could not find %s\n", ALFWaddMediaPath(TEST_OGGVORBIS_FILE));
        ShutdownVorbisFile();
        ALFWShutdownOpenAL();
        ALFWShutdown();
        return -1;
    }

    // (가장 중요한 부분) OggVorbis 스트림을 디코딩하여 Wav PCM 스트림으로 바꾸고
    // OpenAL buffer에 이를 채운 후 buffer ID를 리턴한다.
    uiBuffer = LoadOggVorbis(pOggVorbisFile);
    if (uiBuffer < 0) {
        ALFWprintf("Could not decode %s to PCM.\n", ALFWaddMediaPath(TEST_OGGVORBIS_FILE));
        ShutdownVorbisFile();
        ALFWShutdownOpenAL();
        ALFWShutdown();
        return -1;
    }

    // Generate a Source to playback the Buffers
    alGenSources( 1, &uiSource );

    // 소스와 버퍼를 연결한다.
    alSourcei(uiSource, AL_BUFFER, uiBuffer);

    // Start playing source
    alSourcePlay(uiSource);
    ……
```

8장 예제(PlayOggVorbis) main() (계속)

```
    ……
    // 종료 검사
    while (!ALFWKeyPress()) {
        ALFWprintf(".");
        Sleep( SERVICE_UPDATE_PERIOD );
        alGetSourcei(uiSource, AL_SOURCE_STATE, &iState);
        if (iState != AL_PLAYING)       // Finished playing
            break;
    }
    ALFWGetKey();
    ALFWprintf("\n");

    // Stop the Source and clear the Queue
    alSourceStop(uiSource);

    // Clean up buffers and sources
    alDeleteSources(1, &uiSource);
    alDeleteBuffers(1, &uiBuffer);

    // Shutdown VorbisFile Library
    ShutdownVorbisFile();

    // Shutdown AL
    ALFWShutdownOpenAL();

    // Shutdown Framework
    ALFWShutdown();

    return 0;
}
```


8장 예제(PlayOggVorbis)의 중요 함수들

```
// Try and load Vorbis DLLs (VorbisFile.dll will load ogg.dll and vorbis.dll)
void InitVorbisFile(void)
{
    if (g_bVorbisInit)
        return;

    g_hVorbisFileDLL = LoadLibrary("vorbisfile.dll");
    if (g_hVorbisFileDLL) {
        fn_ov_clear = (LPOVCLEAR)GetProcAddress(g_hVorbisFileDLL, "ov_clear");
        fn_ov_read = (LPOVREAD)GetProcAddress(g_hVorbisFileDLL, "ov_read");
        fn_ov_pcm_total = (LPOVPCMTOTAL)GetProcAddress(g_hVorbisFileDLL, "ov_pcm_total");
        fn_ov_info = (LPOVINFO)GetProcAddress(g_hVorbisFileDLL, "ov_info");
        fn_ov_comment = (LPOVCOMMENT)GetProcAddress(g_hVorbisFileDLL, "ov_comment");
        fn_ov_open_callbacks = (LPOVOPENCALLBACKS)GetProcAddress(g_hVorbisFileDLL,
                                                    "ov_open_callbacks");
        if (fn_ov_clear && fn_ov_read && fn_ov_pcm_total && fn_ov_info && fn_ov_comment &&
            fn_ov_open_callbacks)
            g_bVorbisInit = true;
    }
}

// 동적으로 로드했던 vorbisfile.dll을 내린다. ogg.dll과 vorbis.dll도 같이 자동으로 내려감.
void ShutdownVorbisFile(void)
{
    if (g_hVorbisFileDLL) {
        FreeLibrary(g_hVorbisFileDLL);
        g_hVorbisFileDLL = NULL;
    }
    g_bVorbisInit = false;
}
```

ogg 파일 로딩 & 디코딩 & OpenAL Buffer 생성

```
// ogg 파일을 읽고 디코딩한 후 메모리에 담는다. 그런 후 OpenAL buffer를 생성하고 이 buffer에
// 디코딩한 Wav PCM 데이터를 넣고 이 buffer의 ID를 리턴한다.
ALint LoadOggVorbis(FILE *f)
{
    OggVorbis_File vf;                              // Ogg 파일 객체 정의
    vorbis_info *vi;                                // ogg 파일 정보 객체
    ov_callbacks sCallbacks;
    ALuint uiBuffer;
    unsigned long ulFormat = 0;
    unsigned long ulBufferSize = 0;
    unsigned long ulFrequency = 0;
    unsigned long ulChannels = 0;
    unsigned long ulTotalSamples = 0;
    unsigned long ulBytesWritten;
    char *pDecodeBuffer;

    sCallbacks.read_func = ov_read_func;
    sCallbacks.seek_func = ov_seek_func;
    sCallbacks.close_func = ov_close_func;
    sCallbacks.tell_func = ov_tell_func;

    // Create an OggVorbis file stream
    if (fn_ov_open_callbacks(f, &vf, NULL, 0, sCallbacks) != 0) {
        ALFWprintf("LoadOggVorbis: Input file does not appear to be an Ogg bitstream.\n");
        fclose(f);
        return -1;
    }

    vi = fn_ov_info(&vf, -1); // ogg 파일 정보를 얻는다.
    ……
```


ogg 파일 로딩 & 디코딩 & OpenAL Buffer 생성 (계속)

```
    ……
    ulTotalSamples = (unsigned int)fn_ov_pcm_total(&vf, -1); // 샘플링 전체 개수
    ulChannels = vi->channels;                              // 채널 수
    ulFrequency = vi->rate;                                 // 초당 샘플링 횟수
    ulBufferSize = ulTotalSamples * ulChannels * 2;         // 압축 해제 후의 PCM 데이터 크기
    // 주의: The Buffer Size must be an exact multiple of the BlockAlignment ...
    ulBufferSize -= (ulBufferSize % (ulChannels * 2));
    if (ulChannels == 1)
        ulFormat = AL_FORMAT_MONO16;
    else if (ulChannels == 2)
        ulFormat = AL_FORMAT_STEREO16;
    else if (ulChannels == 4)
        ulFormat = alGetEnumValue("AL_FORMAT_QUAD16");
    else if (ulChannels == 6)
        ulFormat = alGetEnumValue("AL_FORMAT_51CHN16");

    if (ulFormat != 0) {      // 압축 해제 후의 음원 포맷을 알아낼 수 있는 경우에만 디코딩 한다.
        // Allocate a buffer to be used to store decoded data for all Buffers
        pDecodeBuffer = (char*)malloc(ulBufferSize);
        if (!pDecodeBuffer) {
            ALFWprintf("LoadOggVorbis: Failed to allocate memory for decoded OggVorbis data\n");
            fn_ov_clear(&vf);
            return -1;
        }

        // Generate some AL Buffers for streaming
        alGenBuffers(1, &uiBuffer);
        ……
```

ogg 파일 로딩 & 디코딩 & OpenAL Buffer 생성 (계속)

```
        ……
        // ogg 파일을 디코딩하여 Wav 데이터를 얻고, 이를 OpenAL 버퍼에 넣는다.
        ulBytesWritten = DecodeOggVorbis(&vf, pDecodeBuffer, ulBufferSize, ulChannels);
        if (ulBytesWritten)
            alBufferData(uiBuffer, ulFormat, pDecodeBuffer, ulBytesWritten, ulFrequency);

        // Close OggVorbis stream
        fn_ov_clear(&vf);

        // 디코딩에 사용했던 메모리 반환
        if (pDecodeBuffer) {
            free(pDecodeBuffer);
            pDecodeBuffer = NULL;
        }

        return uiBuffer;
    }
    else {          // 압축 후 음원의 포맷을 알아낼 수 없으면 오류 처리함
        ALFWprintf("LoadOggVorbis: Failed to find format information, or unsupported format\n");

        // Close OggVorbis stream
        fn_ov_clear(&vf);

        return -1;
    }
}
```


ogg 파일 로딩 & 디코딩 & OpenAL Buffer 생성 (계속)

```
size_t ov_read_func(void *ptr, size_t size, size_t nmemb, void *datasource)
{
    return fread(ptr, size, nmemb, (FILE*)datasource);
}

int ov_seek_func(void *datasource, ogg_int64_t offset, int whence)
{
    return fseek((FILE*)datasource, (long)offset, whence);
}

int ov_close_func(void *datasource)
{
    return fclose((FILE*)datasource);
}

long ov_tell_func(void *datasource)
{
    return ftell((FILE*)datasource);
}
```

OggVorbis 디코딩

```
// Ogg 스트림을 디코딩하여 Wav PCM stream으로 변환한다.
unsigned long DecodeOggVorbis(
OggVorbis_File *psOggVorbisFile,        // 디코딩할 OggVorbis 스트림
char *pDecodeBuffer,                    // 디코딩된 PCM Wav 데이터가 저장될 버퍼 포인터
unsigned long ulBufferSize,             // 디코딩된 PCM Wav 데이터가 저장될 버퍼의 길이(바이트)
unsigned long ulChannels)               // 채널 수(모노(1), 스테레오(2), Quad(4), 5.1(6))
{
    int current_section;
    long lDecodeSize;
    unsigned long ulSamples;
    short *pSamples;

    unsigned long ulBytesDone = 0;
    while (1) {
        lDecodeSize = fn_ov_read(psOggVorbisFile, pDecodeBuffer + ulBytesDone,
                                 ulBufferSize - ulBytesDone, 0, 2, 1, &current_section);
        if (lDecodeSize > 0) {
            ulBytesDone += lDecodeSize;
            if (ulBytesDone >= ulBufferSize)
                break;
        }
        else
            break;
    }
    ……
```


OggVorbis 디코딩 (계속)

```
    ……
    // Mono, Stereo and 4-Channel files decode into the same channel order
    // as WAVEFORMATEXTENSIBLE, however 6-Channels files need to be re-ordered
    if (ulChannels == 6) {
        pSamples = (short*)pDecodeBuffer;
        for (ulSamples = 0; ulSamples < (ulBufferSize>>1); ulSamples+=6) {
            // WAVEFORMATEXTENSIBLE Order : FL, FR, FC, LFE, RL, RR
            // OggVorbis Order            : FL, FC, FR,  RL, RR, LFE
            Swap(pSamples[ulSamples+1], pSamples[ulSamples+2]);
            Swap(pSamples[ulSamples+3], pSamples[ulSamples+5]);
            Swap(pSamples[ulSamples+4], pSamples[ulSamples+5]);
        }
    }

    return ulBytesDone;
}
```

Homework

- ogg 샘플 소스를 이용하여 .ogg 파일 2개를 스트림 재생하시오.
- 제출물
  - Full Comment가 달린 소스코드가 포함된 프로젝트 폴더
  - 프로그램 제작 설명서
    - 여기엔 소스 코드 포함하면 안 됨.
- 제출기한
  - 5월 9일 밤 12시

Q & A