

Sound Programming

제2장 OpenAL 기초



1



Today's Words in Mind

- 간절히 원해야 비로소 얻을 수 있다.

"어떻게 하면 나비가 되죠?"

"날기를 간절히 원해야 돼.

하나의 애벌레로 사는 것을 기꺼이 포기할 만큼 간절하게."

"죽어야 한다는 뜻인가요?"

"그렇기도 하고 아니기도 하지. 겉모습은 죽은 듯이 보여도
참모습은 여전히 살아있단다.

삶의 모습은 바뀌지만, 목숨이 없어지는 것은 아니야.

나비가 되어 보지도 못하고 죽는 애벌레들하고는 다르단다."

트리나 포울러스의 '꽃들에게 희망을'에서 인용한 내용입니다.

현실의 안락함을 포기할 정도로 간절히 원한다면

누구나 원하는 것을 얻을 수 있다고 생각합니다.



OpenAL(Open Audio Library)

- 개요
 - 3D 사운드를 위한 GPL 기반 cross-platform 사운드 라이브러리
 - OpenGL과 API 컨벤션은 같지만 훨씬 간단함.
 - 여러 회사와 개발자가 모인 컨소시엄에서 개발하다, 현재는 Creative Labs 라는 회사에서 주도적으로 개발하고 있음.
- Resources
 - www.openal.org
 - OpenAL 소스, SDK, Sample Codes
 - Showing how to use OpenAL and it's extensions
- 기능
 - 게임 시 필요한 대부분의 3D 사운드는 모두 지원한다고 주장!



OpenAL의 세 가지 기본 객체

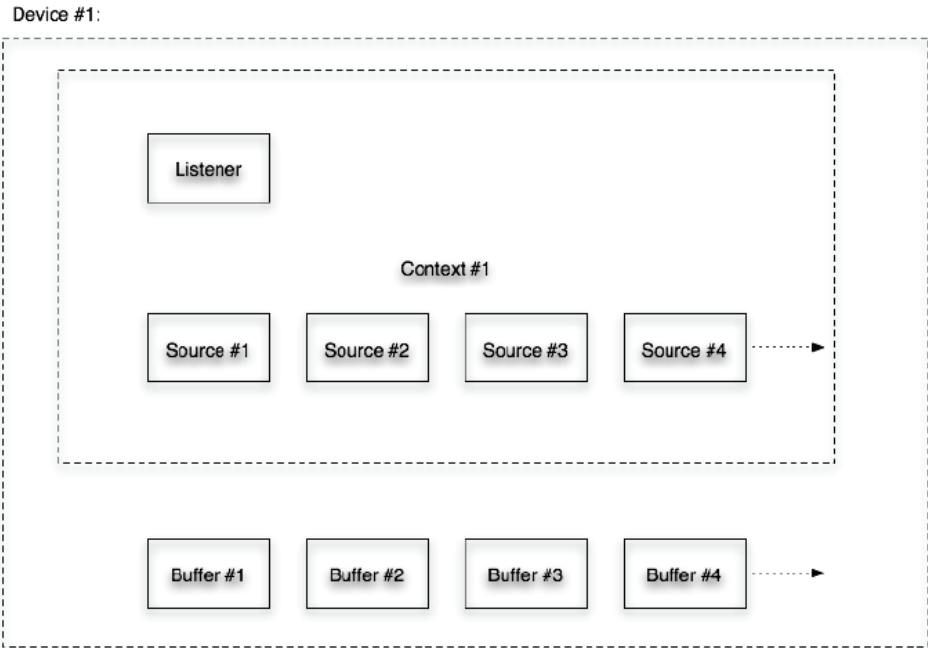
- Buffers, Sources, 그리고 a Listener
 - Buffer
 - PCM sampled data(Wav 파일 내용) itself, Source와 연결된다.
 - Source
 - 3차원 공간에서 위치를 가질 수 있는 음원에 해당. 재생 단위.
 - Listener: Source가 어떻게 들릴 것인가를 결정
 - 오직 하나의 listener만이 가능함
 - 한 Source가 3차원 공간에서 Listener에게 어떻게 들릴 지는 Listener가 Source의 어느 방향/어느 위치에 있느냐에 따라 달라짐.
- **3D Audio World in OpenAL**
 - 하나의 Listener 객체와 원하는 개수의 Source/Buffer 객체를 만들고,
 - Source들과 Listener 간의 위치와 방향을 설정하고,
 - SOrce를 재생하면 완성!



Initializing OpenAL Overview

• OpenAL 초기화

- 최소 한 개 이상의 device 필요
- 한 device는 최소 한 개 이상의 context를 생성
- 한 context는 최소한 한 개 이상의 Source와 한 개의 listener를 생성
- 한 소스에는 최소 한 개 이상의 버퍼가 연결됨



- Buffer는 context에 속하지 않으며, 모든 context들에 의해 공유될 수 있다.



Initializing OpenAL - Device Enumeration

- PC에는 여러 개의 OpenAL rendering device가 있을 수 있음(예: 사운드 카드가 여러 개 있는 경우)
 - OpenAL 응용에서는 어떤 renderer(사운드 카드)를 사용할 지를 선택할 수 있게 해 줌
- OpenAL's Enumeration Extension
 - 프로그래머 또는 사용자가 사용 가능 사운드 장치 목록을 볼 수 있게 해주고,
 - `alcGetString()` 함수 이용
 - 디폴트 사운드 장치의 이름은 무엇인지 알려주는 기능
 - 캡처(녹음) 장치에 대해서도 동일하게 지원함.
 - 프로그래머는 사용 가능한 각 사운드 장치가 어떤 기능들을 지원하는 지에 대해서도 물어 볼 수 있음
 - `alclsExtensionPresent()` 등의 함수 이용



Initializing OpenAL - 초기화 및 종료 순서

- **Initializing**
 - Step 1: Opening a device
 - Step 2: Opening a context on that device.
 - Step 3: Managing OpenAL Objects(listener, sources, buffers)
 - 버퍼 생성: `alGenBuffers()`
 - 버퍼에 Wav 데이터 채우기: `alBufferData()`
 - 소스 생성: `alGenSources()`
 - 소스에 버퍼 연결하기: `alSourcei()`
 - 소스 재생하기: `alSourcePlay()`
 - 3D 재생을 위한 Source와 Listener의 속성 컨트롤: `alGetListenerfv()` / `alListener3f()` / `alSourceci()` / `alGetSource3f`
- **Exiting**
 - Context 제거 - `alcDestroyContext(Context)`
 - Device 닫기 - `alcCloseDevice(Device)`



Initializing OpenAL - Listener 속성

- 모든 Context마다 Listener 한 개가 자동 생성됨
- Listener 속성 설정 및 얻기
 - `alListener()` / `alGetListener()` 함수를 이용
- Listener 속성
 - Master gain - 주 볼륨 역할
 - 3차원 공간에서의 위치 좌표
 - 움직이고 있다면 속도
 - 움직이고 있다면 방향



Initializing OpenAL - Buffer 속성

- **alGenBuffer()**로 버퍼 생성
- **alGetBuffer()**로 Buffer 속성 얻고
- **alBufferi()**로 Buffer 속성 설정
- **Buffer 속성**
 - 버퍼에 있는 음원의 frequency(HZ)
 - 버퍼에 있는 음원의 bit depth
 - 버퍼에 있는 음원의 채널 수(스테레오, 모노)
 - 버퍼에 있는 음원의 크기(바이트 단위)
 - 이 버퍼로 복사된 음원 원본의 위치(사용되지 않음)



Initializing OpenAL - Source 속성

- **alGenSources()**로 Source 생성
- **alSourcei()**로 Source 속성 설정
- **alGetSource()**로 Source 속성 얻기
- **Source 속성들**
 - Pitch multiplier / gain(볼륨) 등 기본 속성들
 - 3차원 음향 처리 관련 속성들
 - 이 소스에 연결되어 있는 Buffer의 ID
 - 여러 버퍼가 큐로 연결되어 있다면 queuing 되어 있는 버퍼 수
 - 재생 위치 관련 정보들



Initializing OpenAL - Queuing Buffers on a Source

- 버퍼 이어 붙이기 기능을 의미
 - 여러 음원을 끊김 없이 연속해서 재생하려면 버퍼들을 소스 한 개에 이어 붙여야 함
- **Buffer Queuing** 방법
 - Source 한 개와 Buffer 여러 개는 앞서 언급한 함수들을 이용해 만들고,
 - Buffer들을 Queue에 붙일 때는 `alSourcei()` 대신 `alSourceQueueBuffers()/alSourceUnqueueBuffers()`를 사용해야 함
 - 큐에서 빠진 버퍼는 재사용될 수 있음
 - 재생은 `alSourcePlay()`를 그대로 사용함
 - 큐에 아직 재생 안 되고 남아 있는 버퍼가 있는 한 재생은 계속됨



Initializing OpenAL - Doppler Shift

- **3D 사운드 효과 중 하나 - 도플러 효과**
 - Source나 Listener가 어떤 속도로 움직이면 소리가 전파되는 정도가 달라짐.
 - 공기 중이냐 물 속이냐 하는 미디어 차이도 있지만, 여기선 공기 중이라고 가정하고 매체는 무시함.
- **OpenAL에서의 도플러 효과 프로그래밍 요소들**
 - `AL_DOPPLER_FACTOR`
 - 의도한 도플러 효과가 미미하거나 너무 클 때 조정하기 위한 **scale** 값
 - `void alDopplerFactor(Alfloat dopplerFactor);`
 - 도플러 효과를 줄지 말지, 준다면 얼마나 크게 할 지
 - `void alSpeedOfSound(Alfloat speed);`
 - 소리의 속력을 설정. 디폴트는 343.3
- 속도 값을 0으로 하면 도플러 효과는 적용되지 않음



Initializing OpenAL - Error Handling

- **alGetError()**
 - 어떤 OpenAL 함수를 호출한 후에 오류가 발생했는지 알기 위해 호출해야 하는 함수.
 - 이 함수는 오류를 검출한 후 오류 값을 리셋하는 역할도 함.
- 사용 예

```
alGetError(); // Clear Error Code

// Generate Buffers
alGenBuffers(NUM_BUFFERS, g_Buffers);
if ((error = alGetError()) != AL_NO_ERROR) {
    DisplayALError("alGenBuffers :", error);
    exit(-1);
}
```



Initializing OpenAL - Extensions

- **OpenAL이 지원하는 확장 메커니즘이란?**
 - 사운드카드 제조사들이 자신들만의 고유 기능을 프로그래밍할 수 있도록 OpenAL에 관련 기능들을 추가할 수 있도록 해주는 기능
- **Creative Labs에서 제공하는 샘플 확장 라이브러리들**
 - EAX(Environmental Audio Extensions)
 - X-RAM: Creative Labs 사가 만든 X-Fi 계열 사운드 카드를 위한 라이브러리
 - Multi-Channel Buffer playback
 - EFX(Effect Extension)
- 프로그래머는 원하는 확장 기능을 **OpenAL이 지원하는** 지를 먼저 검사해야 함
 - **allsExtensionPresent()**
 - **alclsExtensionPresent()**



How to Set Up Your Compiler for OpenAL Programming

- 여기서는 **Visual Studio 2010**을 기준으로 설명함.
 - Visual Studio 2005 이상이기만 해도 no problem.
- 두 개의 파일을 www.openal.org에서 다운로드하여 설치
 - oalinst.exe(OpenAL Installer for Windows)
 - OpenAL 1.1 라이브러리들을 설치함.
 - OpenAL 핵심 기능들과, EAX나 EFX 같은 확장 기능들을 사용하려면 설치해야 함
 - 32-bit and 64-bit 윈도우에서 모두 사용 가능
 - OpenAL11CoreSDK.exe
 - 프로그래밍에 필요한 모든 파일들을 제공
 - 설치 시 설치 폴더를 반드시 기록해둘 것!
 - 예) C:\Program Files (x86)\OpenAL 1.1 SDK\

How to Set Up Your Compiler for OpenAL Programming



After Installing OpenAL SDK

- **Visual Studio 2010** 설정을 위해 알아두어야 할 정보(C:\OpenAL 1.1 SDK\에 설치했다고 가정)
 - 헤더파일 위치
 - C:\OpenAL 1.1 SDK\Include\
 - 라이브러리 파일 위치
 - C:\OpenAL 1.1 SDK\libs\Win32
 - C:\OpenAL 1.1 SDK\libs\Win64
 - 샘플 프로그램 위치
 - C:\OpenAL 1.1 SDK\samples
 - 프레임워크 파일(**OpenAL** 프로그래밍을 도와주는 편의 함수들 관련 소스) 위치
 - C:\OpenAL 1.1 SDK\samples\framework

Sample OpenAL Source Compile & Run - Capture

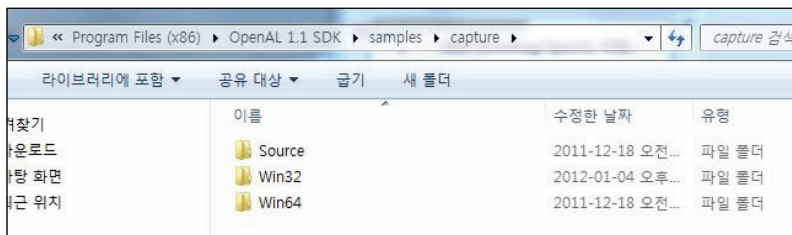
- **OpenAL SDK** 설치 시 같이 설치되는 샘플 소스 중 하나의 컴파일 과정을 따라서 해본다.
- **OpenAL SDK** 설치 폴더가 **C:\OpenAL 1.1 SDK** 라면
 - 샘플 소스는 **C:\OpenAL 1.1 SDK\samples**에 있고,
 - 여기서는 **captures**라는 샘플 소스를 컴파일하고 실행시켜본다.
- **샘플 프로그램 capture** 개요
 - 시스템의 사운드 카드에 캡처(녹음) 기능이 있는지 확인,
 - 캡처 장치 목록을 출력하여 선택하게 하고,
 - 선택된 캡처 장치로부터 2초간의 사운드를 캡처,
 - 캡처한 사운드를 **Wav** 파일로 저장.

Sample OpenAL Source Compile & Run - Capture

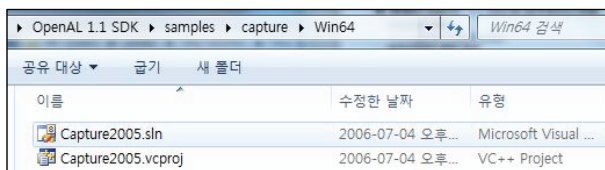


Capture(계속)

- **C:\OpenAL 1.1 SDK\samples\capture** 폴더 내용



- 각자의 시스템에 맞는 **Win32** 또는 **Win64** 폴더로 진입

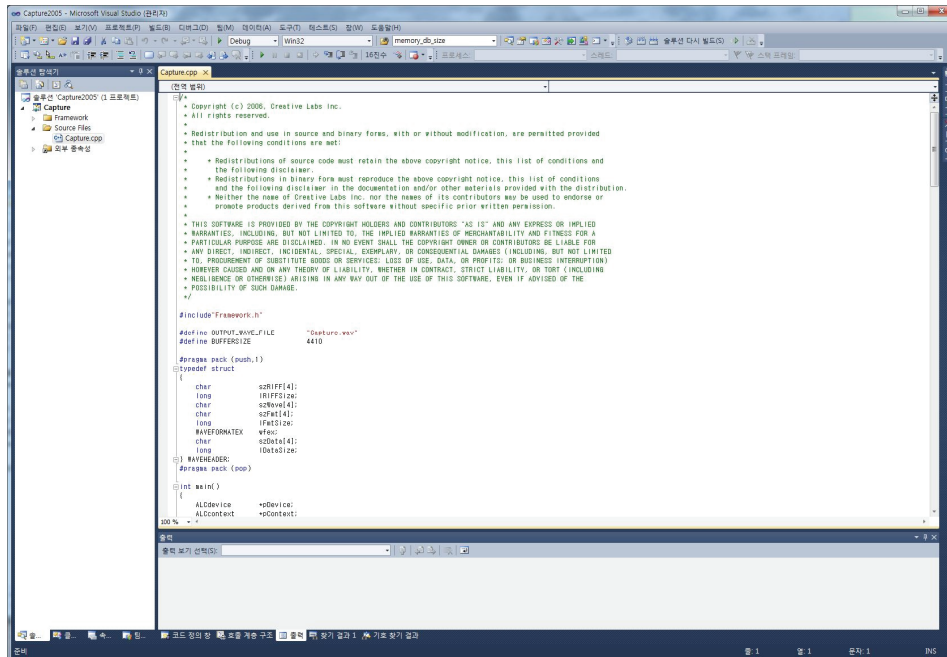


- **Capture2005.sln** 솔루션 파일을 실행
- 이전 버전이므로 2010 버전으로 변환하는 화면이 나타날 것임!
- 변환 마법사에서 "마침"을 눌러 솔루션 버전 변환 완료!



Capture(계속)

- VS2010에서 Capture 프로젝트가 올라온 모습

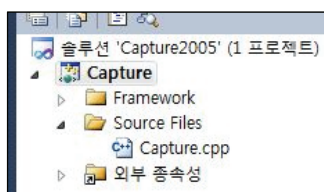


Capture(계속)

- [빌드] - [솔루션 빌드]를 눌러, 또는 F7을 눌러 빌드(즉, 컴파일-링크) 실행
- 컴파일이 성공해도 다음과 같은 경고가 날 것임



- 프로젝트 이름 Capture와 프로젝트 Capture 속성에 지정된 실행 파일 이름이 달라서 생기는 문제임.

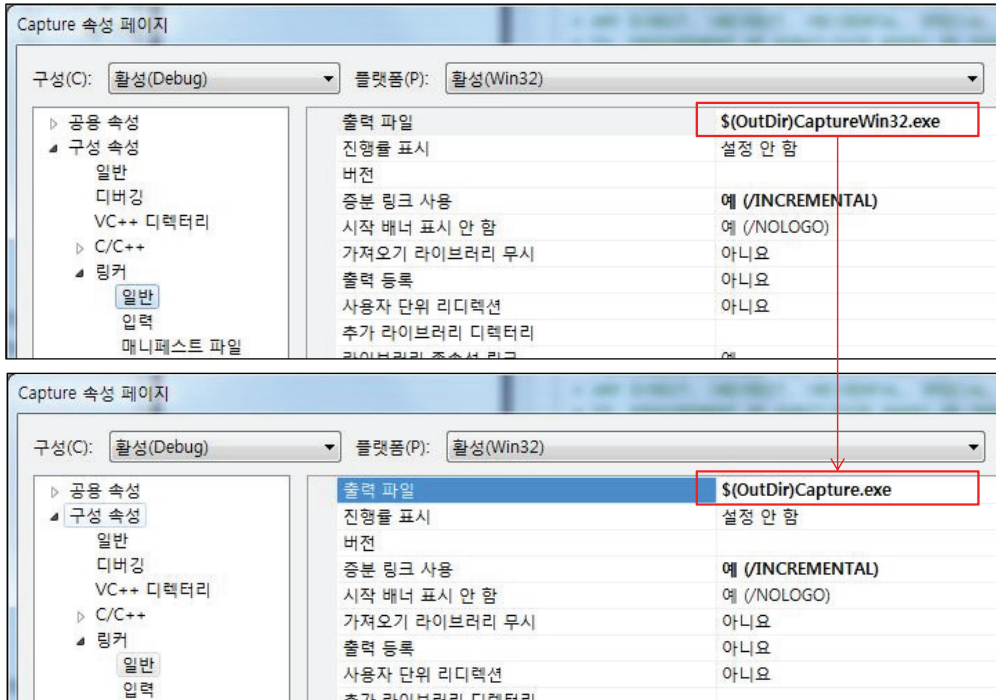


- 프로젝트 속성을 열어 실행 파일 이름을 수정해준다.



Capture(계속)

- [프로젝트] - [Capture 속성] Capture 속성 페이지



Dept. of Multimedia Science,
Sookmyung Women's University



Capture(계속)

- 실행 화면

```

C:\Windows\system32\cmd.exe
Capture Application
Select OpenAL Device:
1. Generic Software<DEFAULT>

Opened Generic Software Device

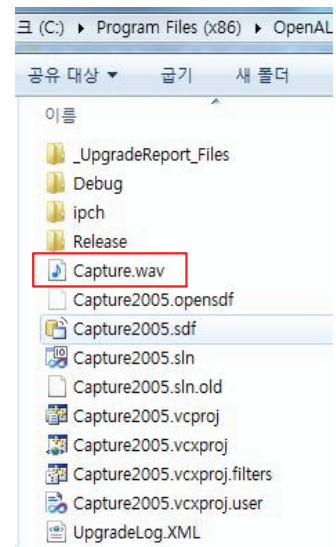
Available Capture Devices are:-
마이크<SB Audigy LS>
라인 입력<SB Audigy LS>
SPDIF 입력<SB Audigy LS>
TAD 입력<SB Audigy LS>

Default Capture Device is '마이크<SB Audigy LS>'

Opened '마이크<SB Audigy LS>' Capture Device

Samples available : 1885
Saved captured audio data to 'Capture.wav'

Press a key to quit
    
```





리포트

- **OpenAL SDK** 최신 버전을 다운로드 하고 설치할 것!
 - 제출할 보고서 없음.
- 설치된 **OpenAL SDK**에 있는 **sample** 중 **Capture**를 **Visual Studio 2010**으로 컴파일 하고 실행할 것!
 - 마이크를 통해 실제 육성을 녹음할 것!
 - 제출할 보고서 없음.
- **OpenAL SDK**에 있는 **samples** 폴더를 각자의 작업 디렉터리로 옮긴 후에도 **Capture** 프로젝트가 컴파일 되게 하라.
 - 힌트: 프로젝트 속성 페이지의 어느 부분을 수정해 주어야 함
 - 2 페이지 이내로 작성하여 이클래스에 제출할 것! 어디를 어떻게 수정해야 하는지 그림으로 표시할 것을 권장함.



Q & A

