Playing & Recording Audio
Audio Queue Services Overview

사운드 콘텐츠 응용 9주차

2010-2
멀티미디어과학과 이종우 교수

Index

Audio Queue Services 란?

- Audio Queue Services
  - 단순하고 오버헤드가 적은 오디오 녹음 및 재생 기능 제공
  - 근본적인 녹음 및 재생 기능을 제공하지만,
  - 고수준 기능을 제공함.
    - 사운드 장치에 대한 지식이 없어도 녹음 및 재생 장치를 사용할 수 있게 해 줌.
    - 코덱이 어떻게 동작하는 지에 대한 세세한 지식이 없어도 복잡한 내부구조의 코덱을 쉽게 사용할 수 있게 해 줌.
  - 몇몇 고급 기능들을 지원함
    - Fine-grained timing control to support scheduled playback and synchronization.
  - 게다가 순수한 **C** 인터페이스로 되어 있음

Audio Queue 란?

- Audio Queue
  - iOS나 Mac OS X에서 오디오 recording 이나 playing에 사용되는 소프트웨어 객체
  - `AudioQueue.h` 에서 `AudioQueueRef` opaque data type으로 정의됨
- Audio Queue 객체는 다음과 같은 일을 함
  - 오디오 하드웨어와의 연결
  - 오디오 처리 과정의 메모리 관리
  - 필요할 경우, 압축된 오디오 포맷 처리를 위해 코덱을 불러옴
  - 녹음이나 재생 작업을 하드웨어와 앱 중간에서 중재함

Audio Queue의 구조

- All audio queues have the same general structure

- 세 부분으로 구성됨

구성	설명
Audio Queue Buffers	오디오 큐 버퍼 각각은 각자 오디오 데이터를 담기 위한 임시 저장소에 해당
Buffer Queue	오디오 큐 버퍼들이 순서대로 나열되어 있는 리스트
Audio Queue Callback	프로그래머가 코딩해주어야 할 콜백 함수들.

Architecture for Recording

- `AudioQueueNewInput` 함수로 레코딩 오디오 큐 생성
- Recording이건 Playback이건 모든 오디오 큐는 하나 이상의 오디오 큐 버퍼를 갖게 되며, 이 오디오 버퍼들이 특정 순서를 이루게 되면 버퍼 큐가 됨.
- Input side
  - 대개 마이크 같은 외부 오디오 장치와 연결된다.
  - 입력 장치로 뭘 쓸 것인가는 사용자가 결정함
- Output side
  - 프로그래머가 작성한 콜백 함수들을 이용한다.
  - 레코딩을 위한 콜백 함수는 새로 입력된 오디오 버퍼들을 오디오 파일로 저장하는 기능을 함.

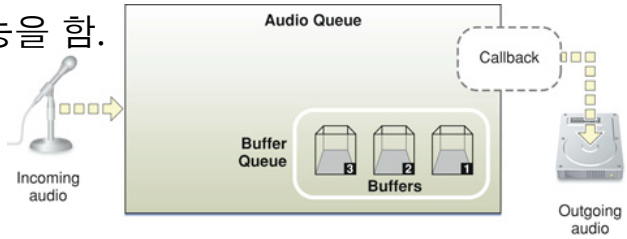


Figure 1-1 A recording audio queue

Architecture for Playback

- `AudioQueueNewOutput` 함수로 재생 오디오 큐를 생성
- Input side
  - 프로그래머가 작성한 콜백 함수가 (디스크로부터) 오디오 데이터를 읽고, 이를 오디오 큐로 건네주는 책임을 짐.
  - 재생할 데이터를 다 보내고 나면 오디오 큐에게 재생을 멈추라는 지시도 콜백이 해야 한다.
- Output side
  - 보통 스피커 같은 외부 오디오 장치와 연결되어 있다.
  - 오디오는 사용자가 지정한 장치(ex: 헤드셋)로 나갈 것임

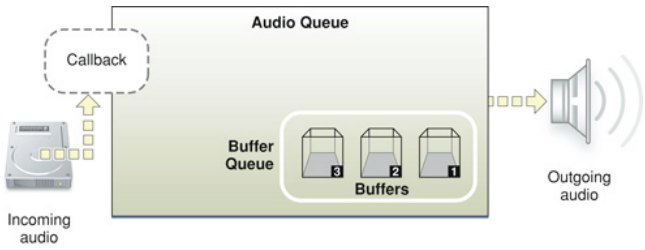
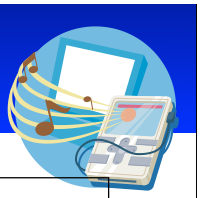


Figure 1-2 A playback audio queue

Buffer Queue와 Enqueueing



- An **audio queue buffer** is
  - a data structure
  - of type `AudioQueueBuffer`
  - as declared in the `AudioQueue.h`

```
typedef struct AudioQueueBuffer {
    const UInt32    mAudioDataBytesCapacity;
    void *const     mAudioData;
    UInt32          mAudioDataByteSize;
    void            *mUserData;
} AudioQueueBuffer;

typedef AudioQueueBuffer *AudioQueueBufferRef;
```

- **`mAudioData`**
  - 현재 재생 중이거나 녹음 중인 일시적 오디오 데이터를 담고 있는 메모리 블록을 가리키는 포인터
  - 나머지 필드들은 오디오 큐가 버퍼를 관리하는 것을 도와줌
- 한 오디오 큐는 여러 개의 버퍼를 가질 수 있으나, 보통은 3개를 가짐
  - 채울 용도, 쓸 용도, 예비용
- 오디오 큐는 자신의 버퍼들을 위해 메모리 관리를 책임짐
  - To allocate a buffer when you call the `AudioQueueAllocateBuffer` function
  - To release buffers `AudioQueueDispose` function

The Recording Process

1. The audio queue fills a buffer with acquired data.

2. The first buffer has been filled. The audio queue invokes the callback, handing it the full buffer (buffer 1).

3. The callback writes the contents of the buffer to an audio file. The audio queue fills another buffer (buffer 2) with freshly acquired data.

4. The callback enqueues the buffer 1 that it has just written to disk, putting it in line to be filled again.

5. The audio queue again invokes the callback, handing it the next full buffer (buffer 2).

6. The callback writes the contents of this buffer 2 to the audio file.

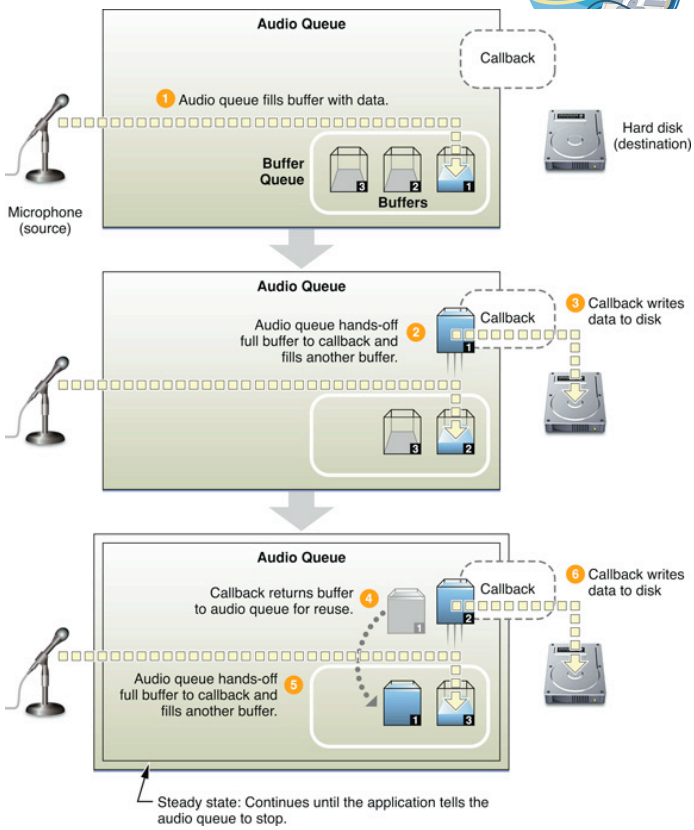


Figure 1-3 The recording process

The Playback Process

1. The application primes(마중물을 붓다) the playback audio queue and invokes the callback once for each for each of the audio queue buffers, filling them and adding them to the buffer queue.

2. Priming ensures that playback can start instantly when your application calls the AudioQueueStart function.

3. The audio queue sends the first buffer (buffer 1) to output.

4. The first buffer has been played, the playback audio queue enters a looping steady state. The audio queue starts playing the next buffer (buffer 2).

5. The audio queue invokes the callback, handing it the just-played buffer (buffer 1).

6. The callback fills the buffer from the audio file and then enqueues it for playback.

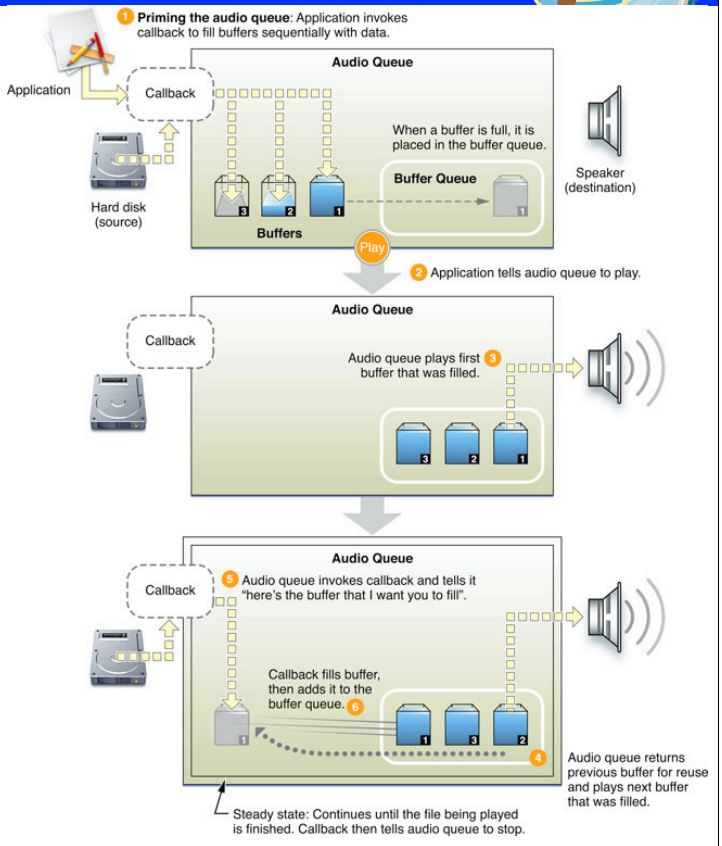


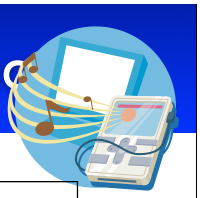
Figure 1-4 The playback process

The Audio Queue Callback Function

- 레코딩 또는 재생 중에
  - 오디오 큐는 필요에 의해 지속적으로 오디오 큐 콜백 함수를 호출함.
  - 호출 간격은 오디오 큐에 있는 버퍼들의 용량에 따라 달라지며, 대개 1/2 초에서 몇 초 간격으로 호출된다.

- 오디오 큐 콜백 함수가 반드시 해주어야 할 의무는
  - 오디오 큐에게 오디오 큐 버퍼들을 제공해주어야 한다는 것.
  - 콜백 함수에서 오디오 버퍼 큐 맨 뒤에 오디오 큐 하나를 추가할 때 호출하는 함수는 `AudioQueueEnqueueBuffer` 함수임

The Recording Audio Queue Callback Func



```
AudioQueueInputCallback (

    void                                *inUserData,
    AudioQueueRef                       inAQ,
    AudioQueueBufferRef                 inBuffer,
    const AudioTimeStamp                *inStartTime,
    UInt32                              inNumberPacketDescriptions,
    const AudioStreamPacketDescription *inPacketDescs

);
```

Parameter	Description
`inUserData`	• 오디오 큐와 그 버퍼들의 상태 정보를 담기 위한 커스텀 구조체를 가리키는 포인터 • 대개 오디오 파일 객체(AudioFileID type)나 audio data 포맷 정보를 포함
`inAQ`	• 이 콜백 함수를 호출한 오디오 큐
`inBuffer`	• an audio queue buffer. 파일로 저장할 새로운 데이터를 담고 있다. • 데이터 포맷은 커스텀 구조체에 지정된 포맷(`inUserData` parameter)으로 되어 있어야 함.
`inStartTime`	• 버퍼 안에 있는 첫 샘플링 데이터의 샘플링 시간. • for basic recording, callback doesn't use this parameter.
`inNumberPacketDescriptions`	• the number of packet descriptions in the `inPacketDescs` parameter.
`inPacketDescs`	• set of packet descriptions corresponding to the samples in the buffer.

The Playback Audio Queue Callback Funct

```
AudioQueueOutputCallback (

    void                    *inUserData,
    AudioQueueRef           inAQ,
    AudioQueueBufferRef     inBuffer

);
```

Parameter	Description
`inUserData`	• a custom structure that you've set up to contain state information for the audio queue and its buffers. • an audio file object(AudioFileID type), audio data format information. • 콜백 함수에서는 이 구조체 멤버를 이용하여 현재 어디까지 읽었는지를 유지하고 있어야 한다.
`inAQ`	• audio queue that invoked the callback.
`inBuffer`	• 오디오 큐가 넘겨준 오디오 큐 버퍼로 콜백 함수는 재생하고 있는 파일에서 다음을 위한 데이터를 읽어 이 버퍼에 채워야 한다.

Using Codecs and Audio Data Formats

- Audio Queue Services는
  - 오디오 포맷 간 변환이 필요할 때 코덱을 활용한다.
  - 설치된 코덱이 있기만 하면, 응용에서는 원하는 코덱이면 무엇이든지 활용할 수 있음
- 동작 원리
  - 각 오디오 큐는 자신만의 오디오 포맷을 가지고 있음(in `AudioStreamBasicDescription` 구조체).
  - When specify the format in `mFormatID` field , the audio queue uses the appropriate codec.

Using Codecs and Audio Data Formats

- A recording audio queue makes use of an installed codec

1. 응용이 오디오 큐에게 어떤 데이터 포맷으로 녹음할 것이라는 것을 알림.
2. 오디오 큐는 녹음한 새 데이터를 코덱을 이용해 변환하고, 콜백 함수를 호출해 변환된 버퍼를 넘긴다.
3. 콜백 함수는 압축된 데이터를 디스크에 저장한다. 이 과정에서 콜백 함수(즉, 프로그래머)는 데이터 포맷에 대해 알 필요가 없다.

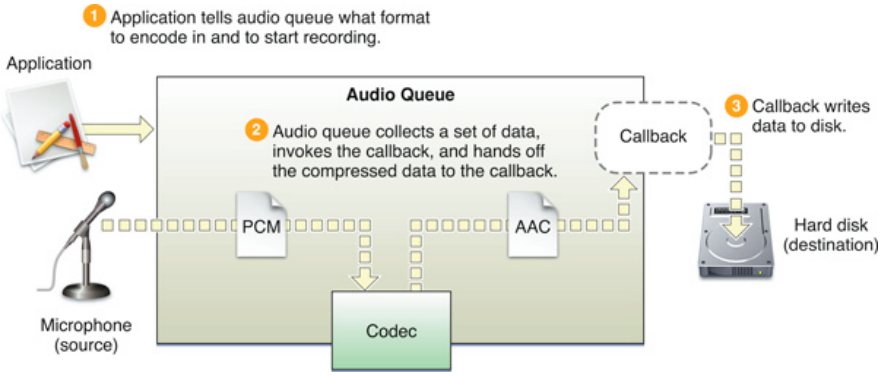


Figure 1-5 Audio format conversion during recording

Using Codecs and Audio Data Formats

- A playback audio queue makes use of an installed codec

1. 응용은 오디오 큐에게 재생할 파일에 있는 데이터가 어떤 포맷인지 알린다.
2. 오디오 큐는 응용에 있는 콜백 함수를 호출하면, 콜백은 오디오 파일을 읽어 읽은 데이터 그대로 오디오 큐에 넘긴다.
3. 오디오 큐는 적절한 코덱을 이용하여 압축해제 한 후 이를 재생 장치로 보낸다.

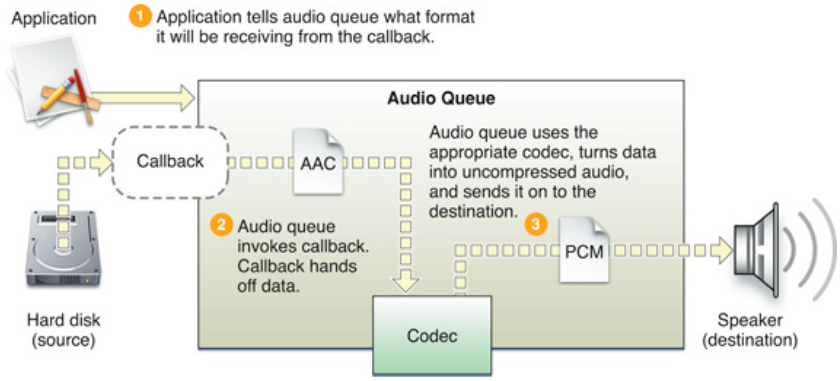


Figure 1-6 Audio format conversion during playback

Audio Queue Control and State

- Audio queue life cycle
  - using 6 functions declared in `AudioQueue.h`

State	Description
Start (`AudioQueueStart`)	• call to initiate recording/playback.
Prime (`AudioQueuePrime`)	• for playback, call before calling AudioQueueStart to ensure that here is data available immediately for the audio queue to play. • not relevant to recording.
Stop (`AudioQueueStop`)	• call to reset the audio queue and to then stop recording/playback. • 동기 or 비동기 mode: ▪ Synchronous stopping happens immediately, without regard for previously buffered audio data. ▪ Asynchronous stopping happens after all queued buffers have been played or recorded.
Pause (`AudioQueuePause`)	• call to pause recording/playback without affecting buffers of resetting the audio queue. • to resume, call the AudioQueueStart function.
Flush (`AudioQueueFlush`)	• call after enqueuing the last audio queue buffer.
Reset (`AudioQueueReset`)	• call to immediately silence an audio queue, remove all buffers from previously scheduled use, and reset all decoder and DSP state.

Audio Queue Parameters

- An audio queue has adjustable settings called **parameters**.
  - each parameter has an enumeration constant as its key.
  - floating-point number as its value.
  - parameters are used in playback, not recording.

- Your application can set audio queue parameters in 2 ways
  - Per audio queue
    - `AudioQueueSetParameter` function
    - lets you change settings for an audio queue directly.
  - Per audio queue buffer
    - `AudioQueueEnqueueBufferWithParameters` function
    - lets you assign audio queue settings that art, in effect, carried by an audio queue buffer as you enqueue it.