Music 3SI: 
Introduction to 
Audio/Multimedia App. Programming

Week #2 - 4/14/2006
CCRMA, Department of Music
Stanford University

So...
• Next week: audio effect >> FFT
• Plug-in: VST >> Max/Pd extern
• GUI: Cocoa > Qt
  ‣ VS on Windows: Ok (no support, though)
  ‣ any volunteer for demo...?
• No Core Image
• Installation help session

That said...
• Not totally discarded
  ‣ sample code provided
  ‣ office-hour help/support available
• Please let me know!

Today...
• Digital audio basics
• Audio API: Stk (RtAudio)
• Stk programming examples
  ‣ signal generator
  ‣ file I/O
  ‣ realtime I/O
• Other APIs
  ‣ PortAudio / ALSA / Core Audio

Digital Audio
• Audio signals stored in a digital format
• Obtained by
  ‣ sampling analog signals
  ‣ creating digital samples by computation
• Audio data as a sequence of samples
Sampling

• Sampling rate (or sampling frequency)
  † determined by sampling interval
  † 2 x upper limit of audio frequency

• Quantization resolution
  † range of numbers for each sample value
  † determines dynamic range (i.e., 16-bit: 96[dB])
  † quantization loss

More About Sampling

See The Difference!

Audio Programming

Audio Programming...

• Create audio sample data
  † by computation, or
  † sampling analog signals: ADC

• Modify the samples, if necessary
  † various effects

• Listen to the data!
  † realtime to DAC
  † wave file

Audio API
**API**

- Application Programming Interface
- Interface provided by OS/lib/app to allow
  - request for service, and/or
  - data to be exchanged
- Audio API: API for audio
  - access to / control of audio hardware
  - sound file handling
  - sound source / effect

---

**Our API: Stk**

- Synthesis Tool Kit
  - by Gary Scavone & Perry Cook
- C++ classes for
  - DSP and sound synthesis (with cool demos)
  - realtime ADC/DAC access (powered by RtAudio)
- Features
  - open source
  - cross-platform

---

**Stk for OS X: StkX**

- OS X framework of Stk
  - dynamic (shared) library +
  - headers
  - other resources (i.e., rawwave data)
- Easier Stk programming
- [http://ccrma/~woony/software/stkx/](http://ccrma/~woony/software/stkx/)

---

**Audio Programming With Stk**

**Sine Wave Data (C)**

```c
#include <stdio.h>
#include <math.h>
int main (int argc, char* const argv[]) {
    int i;
    for ( i=0; i<44100; i++ ) {
        float y = sin( 2 * M_PI * 440.0 * i / 44100.0 );
        printf("y[%d] = %f
", i, y);
    }
    return 0;
}
```

---

**Sine Wave Data (C++)**

```cpp
#include <iostream>
#include <math.h>
int main (int argc, char* const argv[]) {
    int i;
    for ( i=0; i<44100; i++ ) {
        float y = sin( 2 * M_PI * 440.0 * i / 44100.0 );
        std::cout << "y[" << i << "] = " << y << std::endl;
    }
    return 0;
}
```
**Data To Wavefile**

```c
#include <math.h>
#include <StkX/StkX.h>

int main (int argc, char* const argv[]) {
    FileWvOut output;
    output.openFile( "step_1.wav", 1, FileWrite::FILE_WAV, Stk::STK_SINT16 );
    for ( int i=0; i<44100; i++ ) {
        StkFloat y = sin( 2 * M_PI * 440.0 * i / 44100.0 );
        output.tick( y );
    }
    return 0;
}
```

**Data To DAC**

```c
#include <StkX/StkX.h>

int main (int argc, char* const argv[]) {
    RtWvOut dac;
    WaveLoop input;
    char* waveName = argv[1];
    float frequency = atof( argv[2] );
    float length = atof ( argv[3] );
    input.openFile( Stk::rawwavePath() + waveName, true );
    input.setFrequency( frequency );
    for ( int i=0; i<(44100*length); i++ )
        dac.tick( input.tick() );
    return 0;
}
```

**Data From DAC**

```c
#include <StkX/StkX.h>

int main (int argc, char* const argv[]) {
    FileWvOut output;
    RtWvIn adc;
    char* waveName = argv[1];
    float length = atof ( argv[2] );
    output.openFile( waveName, 1, FileWrite::FILE_WAV, Stk::STK_SINT16 );
    for ( int i=0; i<(44100*length); i++ )
        output.tick( adc.tick() );
    return 0;
}
```

**Realtime... Really?**

**So Far: Blocked I/O**

- Program had to “pause”
- Sound length limited
- Unlimited: infinite loop :(
- Any better way to go...?

**Answer: Callback**

- Define a specific function to compute audio
  - called callback function
- Let the audio system call this function when more I/O data can be accepted by the hardware
  - invoked automatically by the audio system
- Referred to as a callback scheme
Callback sample

Assignment

Environment

Machine Setup

- CCRMA Macs: ready to go
- CCRMA Linux boxes & your machines:
  - installation instructions will be provided
- Installation help hours
  - Mon (4/17): 12~2 pm
  - Tue (4/18): 11~1 pm
  - or by appointment (woony@ccrma)

Next Week On Music 3SI

- More on callback: duplex mode
  - simultaneous I/O
- Error handling
- Stk instrument / effect classes
- Audio effect / DSP basics
- (Fast Fourier Transform)
  - (FFTW / Altivec)