Java Music Systems
Columbia University, Prentis 313
Fall, 2007 -- G6610X
Nick Didkovsky, didkovn@mail.rockefeller.edu

Websites:
http://www.algomusic.com (JMSL – Java Music Specification Language)
http://www.softsynth.com (JSyn – Java music synthesis API)
Lecture notes will be posted at http://www.didkovsky.com/JavaMusicSystems

Books

Computer Music. Synthesis, Composition, Performance, by Charles Dodge and Thomas Jerse.
Schirmer Books, ISBN 0-02-873100-X

Meta-Hodos and Meta Meta-Hodos : A Phenomenology of 20th Century Musical Materials and an Approach to the Study of Form, by James Tenney, Frog Peak Music, ISBN: 0945996004

You also need a book on Java programming. You may find one of your own or choose one of the following:

Java Programming: From the Beginning
by K. N. King
Has received excellent reviews. Assumes no prior programming knowledge.

Thinking in Java, by Bruce Eckel
Prentice Hall, ISBN 0-13-659723-8
Full text, updates, and code available at http://www.bruceeckel.com
For the intermediate user, or Java beginner with some programming background.

Java **2** for Dummies, by Barry Burd
While I cannot personally recommend this book, a former student of mine does. Careful to distinguish this book from Java for Dummies by Aaron Walsh which got miserable reviews!

IMPORTANT: Final Project must have an **accompanying paper** submitted in PDF, MSWord, or RTF format! ***The due date of the paper is the same as the project presentation date, which is the last day of class.***

Your final project papers should follow this format:

- Abstract - 3 or 4 sentences describing your project
- Introduction - an expanded, more narrative abstract which also includes your motivations and intentions. If your work is online, include the URL here.
- Detailed description - here you dig into the non-technical perspective on your project. What the user does, how the piece reacts if it is interactive, for example. Include screen dumps of your various gui's or whatever else is graphically included in your project (Wire patches for example)
- Technical description - without resorting a bunch of Java source code, discuss your algorithms and structures. For example, "Every time it repeats, the MusicShape playing the main theme modifies itself by adding new elements, then time compressing all its durations so the overall duration is 8 seconds. I created a Playable to accomplish this task. The Playable was added to the MusicShape's repeat playables. The algorithm chooses a random integer between 1 and 8 and adds that many elements to the MusicShape. Duration are chosen from a set of duration (0,5, 0,25, 0,125), pitches are chosen using a 1/F generator, scaling it to the range of bass clarinet. It uses a two-pass algorithm to time compress. On the first pass, it sums the durations in the MusicShape to get the total. Then it computes a scaler and scales the duration of each element so that the sum remains 8 sec". So you see here that this is a technical description without needed source code. If you need source code, include it.
- Artistic analysis - here's where you write a critique of your work. Did it do what you wanted it to? Why, why not What directions does it point to for the future? What surprised you? For example, "I was puzzled why pitches tended to cluster around the average pitch of the theme after a while. Upon further inspection I discovered that my melody modifier behaved like a band pass filter centered on the mean. I loved this / I hated this. etc"
- conclusion - wrap it up and point to the future. Include a discussion of the role the Java music technologies played in realizing your idea.
- bibliography and sources

Sep

4 Introduction to JMSL and JSyn
Demonstrations, examples, history, and overview. Eclipse demo. Get Eclipse!

11 Introduction to Java
Java Fundamentals, designing GUI's and handling user events
Homework 1: Event handling applet.
Extra Credit: note to pitch applet.

18 More Java
Designing GUI's and handling user events, hard disk organization
Extra Credit: random drawing

25 Introduction to JSyn
Overview of unit generators, circuits, connecting units, getting sound out.
Wire: visual patch editor for JSyn
Homework 1 DUE
Homework 2: Use Wire to design a JSyn SynthNote, deploy in applet with SoundTester.
Extra Credit: Design FM pair with hardwired Fc:Fm and ModIndex ports, deploy in applet.

Oct

2 More JSyn
Envelopes, using JSyn's event buffer to schedule events
Extra Credit Homework: Polytimbral polyrhythms

9 Introduction to JMSL
Hierarchies, and scheduling, using JSyn and JMSL together
Homework 2 DUE
Homework 3: Algorithmic melody using MusicShape and JSyn SynthNote

16 JMSL sonification of musical data
MusicShape, Instruments and Interpreters
Extra Credit Homework: Design a MandelMusic instrument

23 JMSL Score, part 1
JMSL's Music Notation package
Designing JSyn circuits that can be imported into JMSL Score
Designing algorithmic transformations of user-selected notes.
How JMSL's plug-in api works and how to use it.
Homework 3 DUE

30 JMSL Score, part 2
JMSL's Music Notation package
Notating algorithmically generated music using addNote() and using the transcriber
Notating and routing signal processing synth circuits in a score
Homework 4: Compose a piece in JScore either algorithmically or by hand, with custom SynthNotes.

Nov

6 Election Day, school holiday

13 JMSL Real-time performance ***Final Project Proposals Due***
Players, Midi Output, Midi Input, making decisions based on real-time performance data.
Extra Credit: Design a SynthNote and control it via MIDI input

20 More JSyn Instrument design
WaveShaping, Chebyshev Polynomials, playing a sound file, waveshaping a sound file.
Homework 4 DUE

27 Network music, Algorithmic Music Composition
Useful algorithms, mapping mathematical processes to musical events, organizing music algorithmically, recursion, using TCP/IP to create networked JMSL/JSyn pieces, intro to Transjam (Phil Burk's general purpose server)

Dec

4 ***Final Project Presentations***