The Queen's University of Belfast

BSc in Music Technology

Module Description

Module Title:	**Computer Programming for Musical Applications III**
Module Code:	**MTE3007**
Module Convener:	**Gary Kendall (g.kendall@qub.ac.uk, Ex. 4445)**
Teaching Assistant:	**Nick Gillian (ngillian01@qub.ac.uk)**

Teaching Schedule:

Lectures: Monday, 3:00 – 4:30 pm, SARC Multimedia Room
Tutorial: Tuesday, 3:00 – 4:30 pm, SARC BSc Lab

Discussion:

The goal of the module is for each student to be able to program musical applications with interactive audio and with graphic user interfaces (GUIs). To this end, we will be using SuperCollider, not just as an engine for sound synthesis, but also as a programming language. In this respect, SuperCollider serves as a convenient bridge between modern programming languages such as Java or C# and digital signal processing. SuperCollider also provides many special adaptations for music that make it 'relatively' easy to create sophisticated musical applications. During the semester we will examine many of the typical programming issues and methods that are present in nearly all object-oriented languages and in musical applications in specific. We will also be implementing digital-signal-processing algorithms that are typical for audio applications.

SuperCollider is a *free* downloadable program. The version that we will be using in class is for Mac OS X (though version are available for Linux and Windows). You will also be receiving copies of the lectures as SuperCollider documents that you can read and execute. To download SuperCollider:

1. Go to http://supercollider.sourceforge.net/ and follow links under "Downloads."
2. Under the appropriate operating system, download the appropriate release. Mac and Linux users will likely want to select the SuperCollider 3.4 release. Be sure to select the one "With-Extras." The PC version 3.3.1 is admittedly a little behind the other versions, but should be sufficient for this class.
3. On the Mac: Install both the SuperCollider application and the "Optional Installs." From within the "Optional Installs" copy the sc3-plugin-extensions folder to ~Library/Application Support/SuperCollider/Extensions.

Learning Outcomes:

- Students will be able to write and demonstrate computer code for musical applications.
- Students will be able discuss and describe the elements of the programming language.
- Students will be able to describe and implement digital-signal-processing techniques in the programming language.
- Students will be able to write and demonstrate real-time interaction with musical signals.

Assessment:

Programming Assignments 1-12	50%
Programming Handbooks 1-3	45%
Attendance	5%

• Late submissions of the Handbooks will be penalized according to the School's policy.
• Late submissions of programming assignments are governed by rules described below.

Attendance: It is expected that you will attend all classes and participate in discussions. Missing more than three lectures or tutorials during one semester will result in a zero for attendance.

Assignments

I. Programming Assignments

Programming assignments will be given in class each Tuesday and are due by 9 am the following Monday morning. These assignments should be turned in to both the instructor and TA by email. Examples from the assignments of class members will be reviewed each week on Tuesday.

You will receive an email with an evaluation of your assignment on Thursday. The individual sections of each assignment are simply evaluated as 'completed' or not. To be 'completed' the code needs to follow the common conventions of formatting and it needs to work!

After you have received your evaluation, you have one more week to revise and then resubmit any parts of your assignment that have been judged 'not complete.' Revised assignments are due by the following Wednesday morning (along with the first version of the newest assignment). If working code is not submitted by the deadline of the second week, it is recorded as 'not complete' and cannot be resubmitted.

Tutorial assignments will generally have 3-4 parts. Each part will be assessed after the initial due date as either complete, incomplete or not received (C, I, or N).

C: To be complete, all of the assigned elements of the part need to be completed in the submitted work. Omitting elements will lead to an incomplete.

N: If any parts of the assignment are not received by the due date, they are assigned an N and there is no opportunity to redo the assignment.

I: If you receive an incomplete, you will have an additional week to redo that part of the assignment and to turn in a revised version. Revised work files should have the letter 'Rev' appended to the original file name.

The final component mark for the programming assignments as a whole is determined by the number of completed assignments.

II. Programming Handbooks 1-3 (weeks 1-4, 5-8, 9-12)

The Handbook is your summary of the topics covered in the class. Imagine that you are writing this for yourself in the future or for someone like yourself who wants to know the material from the class.

Organize your entries by topics and use subject headings and subheadings to help organize the material. Organization is a large part of your work.

Write a concise summary of each subtopic. Each entry will require an explanation in prose. Write full sentences and complete paragraphs. (Do not write sentence fragments and constantly make lists.) You can copy things such as models of syntax, summaries of arguments and methods from the lecture notes and documentation. Prose should be your own.

Organization. Your Handbook must be divided into the following sections:

I. Elements of the SuperCollider Language.
Information you find useful on SuperCollider statements. Each element of the language should be covered by an entry. Each of these entries should provide a model of the syntax and a prose explanation of element. You will usually want to include short examples of code. This section should be organized and grouped into broad categories with subcategories.

II. The SuperCollider Classes.
Information on the standardized classes bundled in SuperCollider. This includes basic classes like SimpleNumber and Array and extends to all GUI classes. There should be a prose explanation of the essential properties of each class. You can copy and list information on class variables and methods from lectures notes and documentation. Once again, this section should organized and grouped into broad categories with subcategories. Your task is both organization and explanation.

III. Digital Signal Processing.
Summaries of unit generators and DSP algorithms discussed in class. The unit generators do not need to be discussed in the same way that classes are discussed in section II. The focus is on the signal-processing aspect of the unit generators and that is what needs to be described. DSP algorithms created with these unit generators are also organized and described here.

IV. Diary. Personal reflections (at least one for every week) that capture the evolution of your reactions to and ideas about the class and programming. This is also the place to provide feedback (positive and negative) to the instructor on how well the course is serving you. Organize the Diary by the date of the entry.

Work on each Handbook should be accumulative. This kind of assignment preferences consistent day-by-day, week-by-week work. Don't even think about putting it off. Get started today.

Format: The Handbook text should be created with SuperCollider. This has the advantage that any and all examples can be executed immediately within the document.

Sources: Feel free to use any text or images from the lecture slides without references. If you quote something from the documentation, be sure to use quotation marks and to indicate the source with an insertion such as "(SC Help: Functions)." If you use sources from elsewhere be sure also to identify your sources "(Wikipedia: 'Object-oriented Programming')." It is a serious academic offense to represent other people's work as your own.

Due Dates. The Handbooks should be turned in on the following dates:

Week 5, Monday, October 25
Week 10, Monday, November 29
Assessment Period, Tuesday, January 18

On-line Materials.
Lecture files and programming assignments will be available on-line. Visit: http://www.garykendall.net/classes/ProgrammingIII/ for these.

Course Schedule

I. SuperCollider (Week 1-4)
Topics: Overview of SuperCollider and Object-Oriented Programming

IIa. Synthesis I (Week 5-6)
Topics: Signal generation and processing with SuperCollider

IIb. Graphic User Interfaces (Weeks 7-8)
Topics: GUI Classes, GUI design, and interactive programming

IIIa. Synthesis II (Week 9-10)
Topics: Reverberation Algorithms, Convolution

IIIb. Writing Classes (Week 11-12)
Topics: Writing Classes