**Ramapo College of New Jersey**

School of Contemporary Arts

MUSI 650 - 20 : Creative Musical Coding

Fall 2025

Daniel Fishkin

Assistant Professor of Music (Production)

Office: H-108

Phone: (201-684) x7233

Email: [dfishkin@ramapo.edu](mailto:dfishkin@ramapo.edu)

Office Hours: Wednesday (by appointment) Thursday 1–3:30pm

**Class Github:**

[**https://github.com/dfict/MUSI-650-20**](https://github.com/dfict/MUSI-650-20)

SYLLABUS IS SUBJECT TO REVISION DURING THE SEMESTER

*Course Description:* This course provides a foundational introduction to music programming languages. SuperCollider, Max, Arduino, p5js and straight C will be explored through hands-on and creative applications. Students will gain experience with interactive visual art, audio/music signal processing, and machine learning.

This course does not provide a thorough introduction to any single computing language. Rather, it expects and encourages a humanistic approach translating higher level concepts of art to lower and very low levels of applications specific to your intent.

Course Instructor: Daniel Fishkin

*Lecture Time:* Wednesdays, 6:05 pm — 9:35 pm

*Lecture Location:* Electronic Music Lab, H-Wing 204 or remote.

*Instructor:* Daniel Fishkin

*Office Hours:* **by appointment (IRL or Remote)**

*E-mail:* [daniel.fiction@gmail.com](mailto:daniel.fiction@gmail.com)

dfishkin@ramapo.edu

*Materials Fee:* $50

*Textbooks: Bruno Ruviaro, A Gentle Introduction to Supercollider* [*https://ccrma.stanford.edu/~ruviaro/texts/A\_Gentle\_Introduction\_To\_SuperCollider.pdf*](https://ccrma.stanford.edu/~ruviaro/texts/A_Gentle_Introduction_To_SuperCollider.pdf)

Valle, Andrea. *Introduction to Supercollider. Logos Verlag Berlin. 2016. 3832540172*

Shiffman, Daniel. *The Nature of Code. Creative Commons. 2012. 0985930802*

Software **Max**

<https://cycling74.com/>

Max is an object-oriented programming language that makes easy prototyping of sound environments/instruments accessible without needing to learn to code on levels like JS or C. It is a type of coding language in its own way, but the coding is done primarily with virtual representations of wires. It’s very fun and many of my class demos will get you excited about what’s possible—download it and see what you can get working off the bat!

*Note: you can try Max for free for 30 days, and thereafter you’ll need to purchase it, or sign up for a monthly or yearly student subscription. It is affordable. Conn College has Max on every machine in the computer lab!*

**Max comes with an extensive body of educational materials, tutorials, help files, and project examples that can easily be copy/pasted into your own sketches.** Students will be expected to read and study Max / MSP tutorials on their own and learn the syntax of the software throughout the course of the semester!

**Enter the MAX file browser and search:**

**collection:"Tutorials/Max Tutorials@cycling74"**

**collection:"Tutorials/MSP Tutorials@cycling74"**

**Supercollider**

<https://supercollider.github.io/downloads.html>

Supercollider is a platform for audio synthesis and algorithmic composition, used by musicians, artists and researchers working with sound. It is code-based, completely free, light in size, and a little hard to work with.

**Spear**

<https://www.klingbeil.com/spear/>

Spear performs Fourier Analysis in sound files. This is free software that allows you to turn recordings into individual sine waves and edit/select them.

**Audacity**

<https://www.audacityteam.org/>

Audacity is a basic editing program that is excellent for transforming individual sounds.

**Python**

Extremely powerful code based platform

<https://www.python.org/downloads/>

**Csound**

[*https://github.com/dfict/clawyer*](https://github.com/dfict/clawyer)

**Arduino (Processing)**

A yellow and white flower with a white flower and black background

Description automatically generated with medium confidenceHardware coding 101!

Hardware: Daisy ElectroSmith

<https://electro-smith.com>

Small low powered microcontroller

*“Daisy is an embedded platform for music. It features everything you need for creating high fidelity audio hardware devices. Just plug in a USB cable and start making sound! Programming the Daisy is a breeze with support for a number of languages including C++, Arduino, and Max/MSP Gen~. To get started, simply upload an example program over USB, and start tweaking!”*

Flying Fish PCB (DFishkin)

<https://github.com/dfict/flying_fish>

Flying Fish is a scrappy simple circuit using the venerated Atmega328

*Course Repository*: <https://github.com/dfict/MUSI-650-20>

Since this is a coding class, we use Git for version tracking. I am constanting updating, refining, refactoring, and revising code bases throughout the semester and many times throughout the week. Please learn to use **Github!** Download Github Desktop and make an account asap.

Course Objectives:

● Production Techniques: evaluate and apply advanced techniques in music production, synthesis, recording, programming, editing, and mastering;

● Theoretical Foundations: identify and interpret core theoretical foundations of audio;

● Context: analyze and articulate the importance of context in diverse practices within music technology;

● Musicianship and Production: integrate traditional, non-Western, and experimental approaches to musicianship and composition with music production skills;

● Collaboration: collaborate with others effectively and demonstrate leadership in professional situations; and

● Innovation: recognize and formulate innovative approaches to music technologies and careers.

**Course Goals**

● Understanding the relationship between computer programming, music, and digital audio

● Developing core proficiencies relevant to algorithmic composition, interface design, computer programming, analysis and signal processing

● Gaining the ability to read, write, and modify computer code related to music.

**Measurable Student Learning Outcomes**

● Comprehend the importance of music programming in improvised and notated contemporary music (Projects)

● Demonstrate the ability to create programs in SuperCollider, Python, Processing and C++ in the production of compositions and performances (Projects)

● Understand the fundamentals of computer programming languages for music and be able to implement them to build original Audio Unit & VSTplugins (Projects)

**Grading Rubric**

Class Participation / Attendance 25%

Etudes / Projects/ 35%

Presentation 15%

Final Project 25%

**Class Participation / Attendance 25%**

Class attendance is mandatory. We work and experiment in class. Discussions, critiques lectures and demonstrations provide the basis for the successful completion of projects, and they are difficult to re-create outside of class. In order to participate, you must be in attendance.

You have **one** unexcused absence permitted for the semester. Your final grade will drop by 4 points for each further unexcused absence. More than four absences will result in a failing grade for the course. You are expected to participate actively in class by asking questions, bringing energy to discussions, and arriving with prepared homework/projects. Independent motivation is expected.

**Etudes / Homework 25%**

Short reading assignments or creative prompts will be assigned each week. Students will be expected to complete them and come to class ready to discuss their progress. You cannot catch up!—they are meant to be part of a weekly practice that is customized to the flow of the class.

**Presentation 25%**

Do a short research presentation on an artist, or an instrumet, or piece of art that inspires you, taking into note how computation / programming is used as an integral part.

It’s not enough to merely talk about a musician you like, and some synths they use.

You have to identify and explain a technological process.

It should be appropriately cited, referenced. You should not just link to Youtube videos and let the video do the talking.

The purpose of this assignment is to aid the development of your final project by creating an opportunity to research something that interests you for your own creative purposes and research agenda.

25 min.

**Final Project 25%**

Make a creative thesis and execute it. The project may include video, audio, and may somehow utilize a computational process. You can use synthesis, sampling. You can also pursue something mentioned in class but not explicitly explored. You can write code for the Flying Fish Board, for a Noise Toy, in Supercollider. You are invited not to merely make a demo—Make a piece! I would even accept a project in Max MSP. You can use your recorded corpus or create a new one. Students may work collaboratively. The project must be proposed and accepted by instructor. We may present these projects at the Tristan Perich concert, on 12/11, pending feasibility and student interest.

**AI Policy**

The use of Machine Learning tools such as ChatGPT and Claude are permitted, generally. In some cases we will explore them deliberately. In some instances you will be discouraged from using these tools. AI represents a sea change for humanity. It also represents a paradigm shift for pedagogy in digital literacy.

I expect you will want to use AI (ChatGPT and image generation tools, at a minimum), in this class. In fact, some assignments will require it. Learning to use AI is an emerging skill—be aware of its limits. If you provide minimum effort prompts, you will get low quality results. You will need to refine your prompts in order to get good outcomes. This will take work.

Don’t trust any code that your AI provides you. You will be responsible for any errors or omissions provided by the tool. AI is a tool, but one that you need to acknowledge using. Please include a paragraph or citation on any assignment that uses AI explaining what you used the AI for and what prompts you used to get the results. Failure to do so is in violation of the academic honesty policies.

*SCHEDULE*

*Class 1 Wednesday 8/27*

*Introduction to Instructor, me, Daniel Fishkin*

*Introduction to Course and to the syllabus*

*Introduction to course topics*

**Introduction to the SuperCollider 3 environment**

**Server and Language**Reading:

A gentle Introduction to Supercollider

<https://ccrma.stanford.edu/~ruviaro/texts/A_Gentle_Introduction_To_SuperCollider.pdf>

Pages 1-22

SC tutorial: Getting Started With SC: sections 1 – 5

<https://doc.sccode.org/Tutorials/Getting-Started/00-Getting-Started-With-SC.html>

**Etude 1 (due 9/3 next week):**

*Go shopping!*

Try out all examples from 2 - SC2-examples\_1.

Or

Go to <http://sccode.org/> Try out at least 10 sound examples.

Select one that you would like to understand better and modify it.

Modify it so it sounds like something you are excited about.

Submit it on github.

**Nice videos to get you started:**

Live Coding in SuperCollider: a Tutorial with Eli Fieldsteel <https://www.youtube.com/watch?v=rlf8XBxLfRM>

<https://www.youtube.com/playlist?list=PLPYzvS8A_rTaNDweXe6PX4CXSGq4iEWYC>

Watch one or two of these each week

*Class 2 Wednesday 9/3*

Reviewing Supercollider syntax

Learning NodeProxy—The simplest and most direct way to recycle and utilize code in instruments, live, in Supercollider.

Learning Synthdef—the standard way to program synths for global programming concepts in SC

Reviewing Etude 1

**Etude 2 Assigned:**

Prepare 3 synthesis tools in Supercollider that work with NodeProxy OR synthdef, building in arguments, potentially for utilizing with the simple gui.

You can use your favorite exercise from the SC examples from week 1 to have something that you know works, to work off from.

Or make something new! ☺

Bring to class these working instruments and prepare by learning how to use them. We will improvise together!

*Class 3 Wednesday 9/10*

Etude 2 Review: class improv!

Interruption: **Introduction to Max Msp**

**Artist Presentation: David Behrman**

*Music with Memory*

A person with curly hair wearing glasses

Description automatically generated

David Behrman is a composer and artist active since the 1960s. Over the years he has made sound and multimedia installations for gallery spaces as well as musical compositions for performance in concerts. Most of his pieces feature flexible structures and the use of technology in personal ways; compositions rely on interactive real-time relationships with imaginative performers. Together with Robert Ashley, Alvin Lucier and Gordon Mumma, Behrman founded the Sonic Arts Union in 1966. He had a long association with the Merce Cunningham Dance Company as composer and performer, created music for several of the Company’s repertory pieces, and was a member of the Company’s Music Committee during its last years. He was a fellow at the American Academy in Berlin in 2016. Audio recordings of his works are on the XI, Lovely Music, Pogus, New World, WERGO, Black Truffle Records and Alga Marghen labels.

Presentation on Behrman’s Interspecies Smalltalk and Leapday Night

*Etude 3:*

**Learning Max by Doing**

For this assignment, *Adapt “Surging”* to *On the Other Ocean*

*Class 4 Wednesday 9/17*

Comparing Max and Supercollider, discussing the affordances of different tools.

(Why Max? Why SC?)

Exploring different Pitch Sensing tools in each program

Introduction to Iteration

Introduction to Gui

Reviewing Etude 3:

What were your solutions?

(Potentially) Artist Talk with David Behrman

Etude 4:

*Class 5 Wednesday 9/24*

Field Trip to NYC for David Behrman’s Concert

Issue Project Room

[https://issueprojectroom.org/event/open-space-fast-david-behrman-fast-forward](https://issueprojectroom.org/event/open-space-fast-david-behrman-fast-forward-frankie-mann-daniel-fishkin-cleek-schrey)

**Wednesday, September 24th at 8pm**, as part of Celebrating David Behrman—a Fall series honoring the groundbreaking composer and 2025 ISSUE Gala honoree—ISSUE Project Room presents an evening of performances that illuminate Behrman’s enduring legacy of collaboration, technological innovation, and influence across generations of experimental music. The program features two works: Runthrough (1967–), originally developed with the Sonic Arts Union and performed on this occasion by friends of ISSUE, **Frankie Mann** with **Daniel Fishkin** & **Cleek Schrey**; and Open Space with Fast (2020–), a richly textural collaboration between **David Behrman** and artistic partner **Fast Forward**.

Etude 5:

Write 1 page concert report.

*Class 6 Wednesday 10/1*

**Sampling in Supercollider and Max MSP**

**Introduction to Playbuf**

**Introduction to Granular Synthesis**

**Introduciton to 2d.wave**

**Etude 6**:

1. make some samples. Sample an instrument, a phrase of conversation, some environmental sounds: prepare a range of samples and try them out with some of the synthesizers.
2. Incorporate a sampler into your favorite Synthdef or Nodeproxy tools thus far. Remove synthesis engine and replace it with Playbuf, BufRd, or Grainbuf. You can “vibecode.” The point is to get it to work somehow.

*Class 7 Wednesday 10/8*

***Soldering Session!***

We begin soldering *Flying Fish*

[*https://github.com/dfict/flying\_fish/*](https://github.com/dfict/flying_fish/)

Flying Fish is a reprogrammable, microcontroller-based synthesizer that runs the Karplus Strong Algorithm in order to produce plucked tones akin to the sounds of a zither. It is commanded by code, an analog input in the form of sunlight control, and a temperature sensor.

**A pair of blue circuit boards

Description automatically generated**

Etude 7:

Make sure you finish your working flying fish instrument. Request office hours time with Professor if needed. Upload a piece of code from the repo and validate whether it works.

*Class 8 Wednesday 10/15*

*Boxing up Flying Fish*

***First batch of Artist Presentations***

*Vibe Coding with Arduino and DSP*

*Getting different Synthesis Algorithms and DSP to work on 8bit Arduino*

[*https://github.com/dfict/bigg\_fish*](https://github.com/dfict/bigg_fish)

*Etude 8:*

1. Upload and change your own code to flying fish sampler sketch. But first you need to prepare your sample, first by saving a 0.5 second wav file as an 8 bit raw file, and then using the python script to encode your own digital array from that small sample. It needs to be 0.5 seconds, otherwise it will crash.

* or use one of the other DSP examples to make whatever sounds you’d like for the instrument. Add a new sketch and make it your own.

*Class 9 Wednesday 10/22*

***Second batch of artist presentations!***

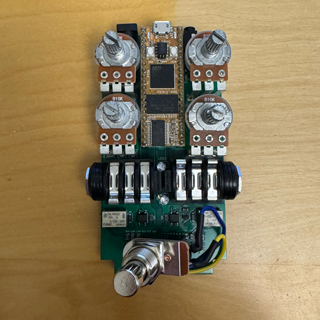
***Soldering day redux!***

*Daisy ElectroSmith*

[*https://github.com/bkshepherd/DaisySeedProjects?tab=readme-ov-file*](https://github.com/bkshepherd/DaisySeedProjects?tab=readme-ov-file)

*We begin making digitally coded guitar pedals.*

*Exploration of Daisy Electrosmith and*



*Etude 9:*

*Get some sounds working on your electrosmith guitar pedal.*

*Class 10 Wednesday 10/29*

*Project / Tinker Day.*

*Etude 10*

***Final Project Proposal.***

*Class 11 Wednesday 11/5*

***Machine Learning Tools in Supercollider and Max***

[***https://www.flucoma.org/***](https://www.flucoma.org/)

The Fluid Corpus Manipulation project (FluCoMa) instigates new musical ways of exploiting ever-growing banks of so und and gestures within the digital composition process, by bringing breakthroughs of signal decomposition DSP and machine learning to the toolset of techno-fluent computer composers, creative coders and digital artists.

These potent algorithms are currently partially available in closed bespoke software, or in laboratories, but not at a suitable level of modularity within the main coding environments used by the creative researchers, namely Max, Pd and SuperCollider, to allow groundbreaking sonic research into a rich unexploited area: the manipulation of large sound corpora. Indeed, with access to, genesis of, and storage of large sound banks now commonplace, novel ways of abstracting and manipulating them are needed to mine their inherent potential.

*Class 12 Wednesday 11/12*

***Gui Revisited: Module.sc in Supercollider***

*Artist Presentation (Ron Kuivila)*

*Class 13 Wednesday 11/19*

*THANKSGIVING BREAK 11/26–11/30 NO CLASS*

*Class 14 Wednesday 12/3*

*Class 15 Wednesday 12/10 or TBA*

*Final Exam*