**Ramapo College of New Jersey**

School of Contemporary Arts

MUSI 415 - 01 : MUSIC, CODE, & INTERACTIVITY

Spring 2025

Daniel Fishkin / dfishkin@ramapo.edu / dfiction.com

SYLLABUS IS SUBJECT TO REVISION DURING THE SEMESTER

*Course Description:* This course provides a foundational introduction to music programming languages. SuperCollider, Python, p5js and Open Frameworks will be explored through hands-on and creative applications. Students will gain experience with interactive visual art, game design, 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, 1:00 pm — 4:30 pm ☺

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

*Instructor:* [Daniel Fishkin](https://dfiction.com/)

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

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

dfishkin@ramapo.edu

course website: <https://github.com/dfict/MUSIC-CODE-INTERACTIVITY>

*always refer here first for up to date information on the syllabus!*

*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*

Software **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)**

<https://www.arduino.cc/>

Hardware coding 101! Arduino is a basic way to interface with sensors, LEDs, motors, electronic components, to make physical changes in the real world

**P5JS**

<https://p5js.org/about/>

<https://p5js.org/reference/p5.sound/>

P5js is the modern update of processing, which was a java based language with similarities to C.

“p5.js is a friendly tool for learning to code and make art. It is a free and open-source JavaScript library built by an inclusive, nurturing community. p5.js welcomes artists, designers, beginners, educators, and anyone else!”

P5js runs in the browser and doesn’t need to compile.

P5 sound library extends p5.js with Web Audio functionality including audio input, playback, analysis and synthesis.

**Sublime Text**

<https://www.sublimetext.com/>

My favorite code editor! Simple, elegant, free.

**Github**

All homework assignments, course materials, etc will be on github!

We won’t use instructure except to link your own submissions as webpages

Please make an account and fork the following:

<https://github.com/dfict/MUSIC-CODE-INTERACTIVITY>

recommended: set up your system with github desktop!

**Then make your own repo for code homework, call it MCI-homework and upload all your assignments there.**

**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!

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 an 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/ 25%

Presentations 25%

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 are permitted one unexcused absence. Your final grade will drop by 6 points for each further unexcused absence. Students arriving more than 5 minutes late will be marked with a half absence. Students who leave early will receive a full absence. More than three 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.

**Artist Presentation 25%**

Do a short research presentation on an artist or piece of art that inspires you. 10 min. 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.

**Final Project 25%**

Make a creative thesis and execute it. The project may include video, audio, and may somehow utilize a computational process to illuminate your corpus. You can use synthesis, sampling. You can also persue something mentioned in class but not explicitly explored. 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 xxxxxxx, 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 1/22*

**Introduction to class materials>**

**Lecture on Creative Coding**

Introduction to GIthub

**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 9/11 next week): Go shopping!**

Try out all examples from 2 - SC2-examples\_1. Go to <http://sccode.org/> Try out at least 10 sound examples. Select one that you would like to understand better and modify it. Submit it on collab.

**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 1/29*

**Etude 1 Review!**

**Lecture Introduction to Foundations of Sound**

**(and if we have time, foundations of modulation)**

**Review of SC tools** (things we didn’t get into last week)

Labtime:

Basics of supercollider and coding covered

Tony Conrad on the Nature of Sound

<https://vimeo.com/73885159>

**Also for Next Week:**

Read: The End of Programming

<https://cacm.acm.org/opinion/the-end-of-programming/>

**Etude 2a assigned:**

Record 5-7 sounds from your life. At least one sound should be a drone—an unchanging sound with pitch components. One should contain a voice. The second file should contain a pitched instrument sound. The third file should contain a percussion sound. All files should not be too short (not less than a second) or too long (not more than a minute). They should not be boring sounds. Don’t record something like typing on your computer. Remember, recording processes used to be physically limited by the parameters of recording technology, represented by the cost of ferromagnetic tape. Don’t waste tape!

Etude 2b

/\*Assignment #2:

Select your favorite example variant and use the Env class to create an envelope so that it plays for a fixed duration and falls silent.

(You will play this in class.)

See github for an example you can draw upon.

You can also write your own synthesis engine, or copy one you like online and thresh out its variables in a nice human to read format.

*Class 3 Wednesday 2/5*

***Synthesis Lecture***

***(decoding automation, FM and AM modulation in Supercollider)***

**Etude 2a Reassigned:**

Record 5-7 sounds from your life. At least one sound should be a drone—an unchanging sound with pitch components. One should contain a voice. The second file should contain a pitched instrument sound. The third file should contain a percussion sound. All files should not be too short (not less than a second) or too long (not more than a minute). They should not be boring sounds. Don’t record something like typing on your computer. Remember, recording processes used to be physically limited by the parameters of recording technology, represented by the cost of ferromagnetic tape. Don’t waste tape!

***Make sure you normalize and convert everything to wav! Label carefully.***

Nodeproxy and gui introductions

**Etude 3 assigned:**

Bring 1-3 “virtual instruments”

Use: synthdefs, nodeProxy, or nDef with Gui machines.

Feel free to use 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 4 Wednesday 2/12*

Going over Joo Won Park’s *Four Hit Combo* (for laptop ensemble)

**Iteration and Buffers**

Going over iteration, for loops, and different classes and methods for filling arrays of data.

Exposure to the powerful concept of iteration in code.

**Etude 4 assigned:**

Bring 1-3 IMPROVED

“virtual instruments”

Use: synthdefs, nodeProxy, or nDef with Gui machines.

Feel free to use 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!

**Put them all in one file and make sure they make sound!**

*Class 5 Wednesday 2/19*

***Nodeproxy Review (with iteration)***

*Building hackable synthesizers with the GUI in supercollider*

***Fourier Analysis in SPEAR***

**Artist Presentations Assigned:**

By next week submit the name of the artist you will Research

**Etude 5 assigned:**

Bring 1-3 “virtual instruments”

Use: synthdefs, nodeProxy, or nDef with Gui machines.

Don’t use the .play method.

Use environment variables a b and c to set up three different instruments.

That’s a soundmaker you can manipulate.

Each should have different arguments that can be controlled through the gui.

Feel free to use 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 6 Wednesday 2/26*

**Buffers:**

**Using Playbuf to play sound files in Supercollider**

**Hacking your synth engines to utilize recording sounds.**

***Hardware Soldering!***

***Etude X Assigned:***

*Corpus Collection:*

Record and prepare a collection of 25 “one shot” sounds (~1second max) through a variety of means.

1. Pick an instrument and record 25 different notes in a gamut of sounds that will be the collection of your piece. For example, you may choose related two pentatonic scales on guitar in two octaves. Or one scale in four octaves. Be creative. Make sure one sound is the “tonal center” or “root note” of the collection.
2. Record a collection of related and unrelated sounds that will be the sound palette of a piece. These, then, would then not have to be the same instrument, but different one shots prepared uniquely. You can use sounds from life, and sounds from a particular instrument. But they don’t have to be organized by pitch.
3. Split the difference between 1 & 2. Use different techniques on an instrument but keeping the pitch groupings

*Make sure you record these sounds well, in the Ramapo Recording studio.*

*Class 7 Wednesday 3/5*

*Buffers Part II*

*Noise Toys Part II*

*Class 8 Wednesday 3/12*

*First Batch of Artist Presentations*

*Living Steams Hackathon due*

**3/16—3/23 SpRING BREAK!**

**No class 3/19(!**

*Class 9 Wednesday 3/26*

*Second Batch of Artist Presentations*

***Introduction to p5js***

***—making sounds in the browser***

[***https://p5js.org/***](https://p5js.org/)

[***https://p5js.org/reference/p5.sound/***](https://p5js.org/reference/p5.sound/)

<https://editor.p5js.org/thomasjohnmartinez/collections/Dp0zGclVL>

some cool sound examples to learn from

[*https://github.com/ogbabydiesal/Living-Streams*](https://github.com/ogbabydiesal/Living-Streams)

***The Musical Web***

***Introduction to tone.js***

***Etude Y Assigned:*** *Living Streams Hackathon*

**Living Streams by Tommy Martinez**

[**https://living-streams.vercel.app/**](https://living-streams.vercel.app/)

[**https://thomasjohnmartinez.com/**](https://thomasjohnmartinez.com/)

Tommy Martinez is an artist and programmer working primarily through research, sound and code. He creates software and musical systems for the internet, embedded devices, and for live multichannel performance. Martinez has performed at MoMA PS1, The DiMenna Center for Classical Music, Fridman Gallery, and Pioneer Works. He has lectured on sound and electronic art at School for Poetic Computation, UC Berkeley, Stanford University, and Kunstakademie Düsseldorf.

As a programmer and systems designer Tommy has worked with artists such as Laurie Anderson, Toni Dove, Nicole Eisenman, Pierre Huyghe, Kristin Lucas, Florian Meisenberg, and Martine Syms. His work as a collaborator and engineer has been exhibited at Artists Space, Asia Society, Bridget Donahue Gallery, Carnegie Mellon University, Henie Onstad Kunstsenter, ICA at Virginia Commonwealth University, Kunsthalle Basel, The Shed, Simone Subal Gallery, and the 2019 Whitney Biennial.

Tommy was previously the Director of Technology at [Pioneer Works](https://pioneerworks.org/) where he led the Tech Residency and other initiatives at the space.  He is currently an Adjunct Professor at NYU’s Interactive Telecommunications Program (ITP) and Integrated Design & Media (IDM) and currently teaching The Musical Web at School for Poetic Computation, a class exploring sound and composition on the internet. Tommy holds an MFA from the [Sonic Arts](https://www.sonicartsmfa.org/) program at CU

*Living Streams* is a generative web composition that runs in the browser, using a collection of 25 recorded sounds.

*Class 10 Wednesday 4/2*

*Class 11 Wednesday 4/9*

*Class 12 Wednesday 4/16*

*Class 13 Wednesday 4/23*

*Class 14 Wednesday 4/30*

*Class 15 Wednesday 5/07 or TBA*

***Final Exam***