Live Coding

A User's Manual

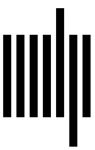
By: Alan F. Blackwell, Emma Cocker, Geoff Cox, Alex McLean, Thor Magnusson

Citation:

Live Coding: A User's Manual
By: Alan F. Blackwell, Emma Cocker, Geoff Cox, Alex McLean, Thor Magnusson
DOI: 10.7551/mitpress/13770.001.0001
ISBN (electronic): 9780262372633
Publisher: The MIT Press
Published: 2022

OA Funding Provided By:

OA Funding from Author



The MIT Press

3 Expositions

Rangga Aji, Alejandro Albornoz, ALGOBABEZ, Rafaele Andrade, Jack Armitage, Pietro Bapthysthe, Lina Bautista, Renick Bell, Ashlae Blum(e), Alexandra Cardenas, Lucy Cheesman, Joana Chicau, Nick Collins, Malitzin Cortes, Mamady Diarra, Claudio Donaggio, Jason Freeman, Flor de Fuego, Sarah Groff Hennigh-Palermo, HAUS++, Mike Hodnick, Timo Hoogland, Miri Kaat, Abhinay Khoparzi, Shawn Lawson, Melody Loveless, Mynah Marie, MicoRex, Fabrice Mogini, Kofi Oduro, David Ogborn, Jonathan Reus, Antonio Roberts, Charlie Roberts, Jessica A. Rodriguez, Iris Saladino, Kate Sicchio, th4, Anne Veinberg and Felipe Ignacio Noriega, Rodrigo Velasco, Elizabeth Wilson, Anna Xambó

Photo by Alieneta Firdausi

Photo by Ahmad Mulkan Karim

Rangga Aji
Program Director: October Meeting—Contemporary Music & Musicians,
Special Region of Yogyakarta, Indonesia

For me, live coding is a method that could be implemented in any kind of artistic practice. I live code in my music performance work and sometimes also for live visuals. I got interested in live coding when I saw it as another possibility in art and music creation, especially given the generative ideas that could be implemented on it. The way of projecting/showing the coding process also adds the variant of presentation, especially in the ontology process of electronic music.

I love to share what I know about live coding practice with people. Sometimes I do little private live coding sessions for individuals or groups, online or off-line, helping with installation and teaching a bit about Sonic Pi, FoxDot, TidalCycles, and Hydra and then Estuary, Troop, and Flok for the possibility of remote live coding. I also made an introduction video about live- coding music and visuals in Bahasa Indonesia.[1] I feel like it still couldn't provide a better explanation of live coding, but I decided to make it happen so people could at least become intrigued enough to try to learn it by themselves.

For me, both live coding from scratch and live coding from prepared code are exciting and promising for my live coding practice. With the first method, I can learn how to directly or slowly decide what kind of musical structure I want to build without depending on any prepared code. With the second method, I can have some control over my music composition while introducing aspects of live coding. I'm also thinking about the possibility of both approaches in exploring the idea of certain musical etude compositions for live coders.

Based on what I've been experiencing in Indonesia, live coding is starting to slowly emerge. Even though only certain people became intrigued and started their own live coding activities in certain places here, I think it will soon become something unexpected, but I'm not sure what.

One thing I've noticed is that live coding often brings up a new topic to be discussed—for example, the idea of remote live coding and an ensemble. I remember a discussion about this with fellow live coder friends from Indonesia and India, Maria Maya Aristya and Abhinay Khoparzi.[2] One of the many things we talked about was the musical possibilities that could be implemented in a remote live coding performance as an ensemble. This is exciting for me since it seems like it could be the opening of further possible perspectives.

https://linktr.ee/ranggapuraji

Alejandro Albornoz, a.k.a. co(n)de Zero, and Christian Oyarzún.
Image credit: Constanza Lobos

Image credit: Alejandra Caro

Alejandro Albornoz, a.k.a. co(n)de Zero
Universidad Austral de Chile,
Valdivia, Chile

For me, live coding is a fresh way to embrace live electronic music, a field I've been involved with since 1989. With my band Arteknnia, we moved from electro-industrial live acts with synths to synth-pop by MIDI sequences, drum machine, and guitar. After 1995 I started to compose just with a synth and cassette tapes, resulting in my ambient/electronic dance music solo project *Mankacen*. From 2003 onward, this was in parallel to my electroacoustic output in a more "academic" compositional path. Whatever the style, I've used different DAWs (digital audio workstations) extensively, which progressively led me to lose interest in performing electronica, focusing my practice on acousmatic. I discovered TidalCycles during a gig by Yaxu (Alex McLean) in Sheffield in 2015. It blew me away, so I've started to learn first ixi lang and then Tidal. I felt that I was in front of a completely new way to perform, something that matched with my incipient ideas on using simple algorithms in a fresh live situation. Before that, I would say since 1999, I was using simple algorithmic procedures to get some materials but, usually, in deferred time.

I think showing the code on your screen is so nice, friendly, and transparent. Something very important for me is to perform from scratch . . . that's so adrenaline pumping and challenging, making me feel alive. There is enough room to mix stable pulsating pattern structures and textural material. . . . TidalCycles in particular allows me to create new electronic tracks and at the same time materials that I use in acousmatic composition. Probably, this would be my only and modest contribution: an unbiased approach in terms of styles or aesthetics, moving between broken post-techno structures, glitchy textures and rhythms, Chilean folklore patterns, and granular synthesis. All of this operates in a recursive personal ecosystem, from which outputs are electronic tracks and acousmatic pieces. Among all this, I've developed a special interest in working with voice and words. However, in spite of the possibility of creating known musical styles, the most relevant aspects for me are the unexpected structures and sounds provided by these techniques.

I lived in Sheffield, UK, between 2015 and 2019, where I started to be in touch with TOPLAP, Tidal Club, Eulerroom, and Algomech Festival, among other live coding instances. Since then, I've returned to my country, Chile, with the intention to spread the practice and tools to as many people as possible. This is an ongoing project that is growing slowly and has been heavily affected by the COVID-19 pandemic. In this endeavor it has been very important to be in touch with people and groups, including live coder and media artist Christian Oyarzún, the cultural center Taller La Cisne Negro, and the Pueblo Nuevo netlabel. Hopefully, we are intensifying our activities this 2021, which includes holding the International Conference on Live Coding (ICLC) in our city, Valdivia. Live coding is still very new here, but I'm convinced it is a great tool, even beyond artistic outcomes. I see it as a way to gather people through open and democratic spaces involving art and technology, especially for children and young people's education.

alejandroalbornoz.wordpress.com/conde-zero/

Photo by Antonio Roberts

ALGOBABEZ (Shelly Knotts and Joanne Armitage)
UK

ALGOBABEZ came together through a shared skepticism of code-bro cultures, a flurry of estrogen, and overloaded algorave lineups. Collaboration was inevitable. We were both performing regularly as solo performers but were drawn together by the maleness of the scene. Our first performance took place in the corporate sheen of the Open Data Institute as part of the Leeds Digital Festival. We felt a sense of disrupting the atmosphere—when we started playing, the crowd changed. The women moved forward; the men moved backward. We had to let rip! Though totally unplanned, this set up a "legend" around ALGOBABEZ as an obnoxiously noisy and unapologetically feminist double act.

Our individual live coding practices meant we naturally fell into specific musical roles: Shelly making textural drone sounds and Joanne tending toward rhythmic material, juxtapositioning Shelly's lo-fi software synth sounds with Joanne's hi-fi MIDI instruments.

Beyond algoraving, we have looked at new ways of knowing and controlling sound through performance (i.e., BabeNodes, Vibez, Chemical Algorave), combining Joanne's knowledge of physical computing and Shelly's skills with biometrics and algorithmic systems. Shelly's experience in networked music has enabled our continued collaboration as the physical distance between us has increased.

We both engage with feminist theory in our academic work, and this has informed the way we present ALGOBABEZ. Connecting *ALGO* to the ironic use of a term of endearment allows us to overtly perform gender in academic and electronic music scenes, where women are underrepresented. We've used this "ironic gender performance" as a way to build narratives around our work. As a band, we've rejected the use of prewritten code and structures—acting in the moment, responding to context, audience, mood, alcohol consumption, and each other. Before performing, we don't discuss structures, preferring to just bounce off each other, developing a structure as we work, continually creating and resolving tensions. We work to strip back dense textures and create space for each other across the frequency spectrum. We work on other computer music projects that are centered around collaborating with, teaching, making space for, and promoting women in the field. One joint project is OFFAL (Orchestra for Females and Laptops), an international telematic women-only laptop ensemble who perform through audio streaming and web interfaces.

The sound world of algorave has changed as a result of ALGOBABEZ taking noisy sounds and seamlessly integrating them into four-to-the-floor beats. We didn't set out to play a particular genre or type of music, but this emerged out of improvisation and performance—we reference the more experimental side of live coding (often concerned with SuperCollider synthesis) and transpose it into the algorave context while projecting ourselves as bad-ass women. We've seen this experimentation with sound propagate across the scene since we started playing.

ALGOBABEZ has also made a significant contribution to diversifying the gender balance in the UK algorave scene through teaching workshops and acting as role models. Through this, we have achieved a critical mass of women who are performing, teaching, and organizing live coding activities.

https://twitter.com/algobbz; https://algobabez.bandcamp.com

Photo by Felice Hofhuizen

Photo by Paulus van Dorsten

Rafaele Andrade

Knurl, an interactive and open-source cello, is a creation of the composer and cellist Rafaele Andrade. It has been reaching audiences all over the world for its nonstandard vision, bringing music into political, technological, and environmental discussions. This project brings open-source practices, sustainable three-dimensional printing, creative coding, and sustainable energy sources into the hands of a diverse team composed of local artists, designers, scientists, and programmers. Besides Knurl, Rafaele explores themes such as climate change, globalization, and feminism in her own artworks; every matter matters, and every message is also part of an ongoing process.

Knurl is a product of creative coding and open-source culture: its interface and web platform were designed by members of Netherlands Coding Live at the instrument inventors' initiative in the Netherlands. During its year of development, it became the product of an artistic research organization (Knurl Lab) that brings creative coding into the hardware of a musical instrument, in dialogue with issues that our music world has been facing (climate change, globalization, and so on).

The intention to unify its creative approach as a single process is clear: to develop an idea through active practice. It is to collapse prototypes, one after another, and collect as much feedback and evaluation as possible. To create is to finish thousands of such cycles as a deep experience into your personal life. Rafaele believes that these experiences can be enhanced by using all your senses, your body, your well-being, and your health.

As science moves forward with the development of new tools, the invention of new musical instruments can also open a new chapter in music history. Realizing that Knurl isn't only an instrument but also an interface, a platform, and a piece of hardware, this project intends to solidify those results, testing their applications as tools that also represent cultural heritage.

New repertoires for the cello and electronics in collaboration with other instruments have been made and presented at international music festivals, conferences, and artistic residences. The intention is to explore how instruments and technologies can create new bounds and approaches between their users, creating new forms of relationships and entertainment. Creating new instruments also provides the opportunity to explore sustainable practices—for example, ways to capture sunlight or even sound output as energy storage, as well as the development of biodegradable materials for acoustics solutions.

Photo by Maggie Kane https://www.instagram.com/streetcat.media/

Jack Armitage
UK

Live coding is a set of tools and methods for sculpting dynamic musical systems via algorithmic notation, often but not exclusively through the medium of text. I first engaged with it as a way to escape the limits of traditional music software, and still do. Since I started I have pursued two goals with my live coding practice. First, I strived to raise live coding to equal status musically with pop and dance produced via traditional means, to humanize coding and inspire "noncoders" to reclaim it from the techno-academic elite. Second, I have used it as a platform to research more richly embodied forms of dynamic musical media, to liberate us from the anachronistic typewriter-and-paper-emulator interfaces that demean us daily. While I have in my view contributed significantly to the first goal, its measure of success is whether I inspire anyone to surpass me, and so far that hasn't happened, but I remain optimistic. The second goal has so far been open-ended and long-term oriented, with only a few short papers and prototypes to speak of.

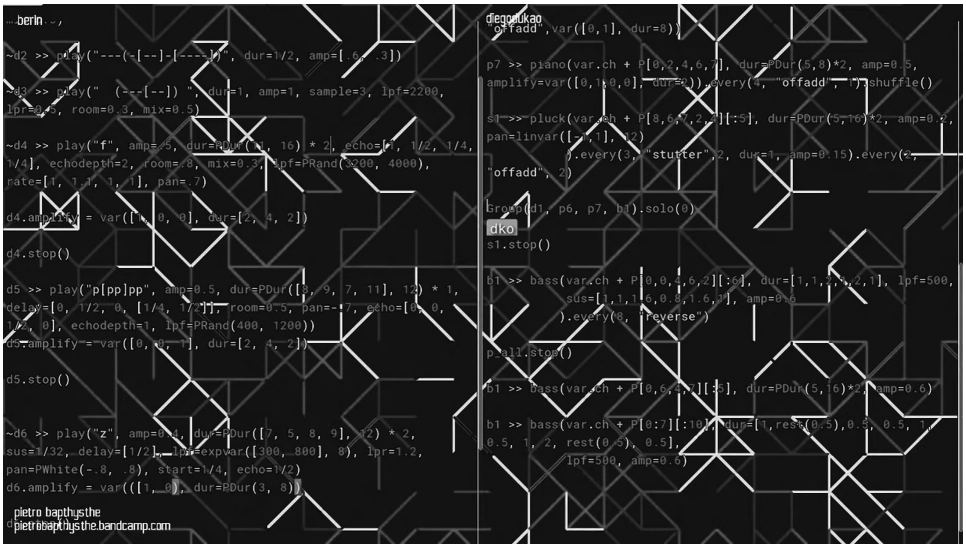
I see my practice as locked in tension and even opposition with other live coders, and I suspect that others feel the same way (about themselves and me). This tension arises when I import mainstream pop/dance references into live coding and similarly when I take liberties with live coding dogma. This is a necessity when performing on pop/dance music lineups where I am the only live coder, but for me it also highlights the frequently self-serious presentation of live coding and its often spartan musicality. At the end of the day, there is always something about other live coding practices to be inspired by; it's especially inspiring to see first-time performers, which is frequent in the live coding community as compared with other music communities I've been part of.

Since my primary goal has been nontechnological and I arrived in the scene at a time when powerful live coding systems had already matured, I have dedicated my system design energy toward the future and lie in wait for the appropriate moment to act. This is another way of saying that I haven't actually built anything myself yet.

Coding in the large is gradually becoming live by default, and the days of coding as a singular practice of moving text using a typewriter are numbered. Consequently, live coding itself as a stand-alone term or field does not have a future: it will not permeate more than it already has, and it will soon be outmoded by younger generations who look and talk differently. However, its impact as a stepping-stone is already in my view historically undeniable. As for the future, I believe that the things that reference live coding (explicitly or not) will be numerous and marvelous.

As it is not a general term, I doubt that live coding means anything to the "general public" other than something technical sounding and thus vaguely threatening. If as an experience it is foisted on an unsuspecting public, audiences initially perceive live coding as a curious academic gimmick with a poor technique-to-music ratio. If they manage to stay through a whole show, they start to see that every performer's approach and music are different and that, rather bizarrely, this means there must be a whole subculture of this stuff going on somewhere—and they would be correct. But in the future, the general public will be much more likely to live alongside the offspring of live coding than the parent.

jackarmitage.com



Computer artwork by Diego Moreira Guimarães

Photo by Mari Moraga

Pietro Bapthysthe (Diego Dukão and Berin)
Brazil

For us, live coding began as another way to explore our musical interests together. Over time, it proved to be a perfect fit because we feel more able to reproduce what we are imagining without having to struggle with the physical limitations of instruments.

We started to explore live coding together as a practice in June 2019, during a Python conference in the northeast of Brazil. Starting with live coding as an improvised way to have a party to celebrate the end of the conference was amazing! It helped us to grow a strong need to promote a community in Brazil. Since then we've been active members in Algorave Brasil, helping newcomers, organizing events and workshops, and sharing content about live coding. During 2020, we also released monthly records with recorded live coding sessions.

The live coding tool we use the most is FoxDot. We've already made a few contributions to the project and to Troop as well. But we also have our own SonicBox, which is a mobile app that interacts with Sonic Pi, allowing us to make improvised melodies with a mobile device, together with the music being played by FoxDot.

Currently, here only people that practice live coding listen to live coded music. It's a small niche—a very prolific one—but small. We think that will change and that live coded music will blend with regular music and that sometimes we won't even know, for example, that rap that is playing was made with live coded music. We also think that schools will adopt live coding to teach programming and music at the same time because it is such a good way to learn both of the subjects! Teachers and students will love it.

A lot of people think they could never play an instrument, and a lot of people think they could never program. So when they hear that you deal with both things together, they think that it could be the most intangible thing ever. But when you show them your live coded performance, they enjoy it or, at least, find it interesting. And when you spend just a few minutes showing them your code and explaining what you're doing, they start to think "Hey, I could learn that."

pietrobapthysthe.bandcamp.com

Lina Bautista performing as Linalab.
Image credit: Iván Paz

Lina Bautista
Spain

For me, live coding is a practice in which the brain has a unique connection with the music and the public. It is a particular way to show your thoughts. When I started with live coding, I was just curious about different performative practices, in search of my way. What I found is that live coding really makes me perform in a singular way. I can almost hear my brain going fast, making decisions and reacting to sounds. Live coding is also about the community. I think it is a safe space to learn, make mistakes, get support from others, and have fun.

I think that my role in the community has been about encouraging people to get into live coding by performing in different contexts and through connecting people. I started TOPLAP Barcelona along with Iván Paz, and since then, the community has been growing with the care and ideas as I learned it: sharing and working together, although in the community we do not all use the same tools or play the same musical genre.

My performing style is basically starting from scratch and using just a few lines of code during the session. In this way I try to make good sessions with a few elements, and I try to make the code simple and understandable so the general public can feel more included, encouraging them to do the same.

I'm aware that live coding is moving faster into the mainstream. We can perceive that in our context. I've worked hard to explain to institutions, festivals, and projects that live coding is not only adding some code to the screen. It is about the way you do things and about a community that works in a particular way. I also see that each time more live coders develop their own tools, languages, and libraries. I see more relationships between hardware and tools, such as the use of artificial intelligence (AI), for example. I think all of that will take live coding to new and exciting places.

Some people just enjoy a session, like an algorave, because it is a kind of geeky party. They feel they are in *The Matrix* surrounded by code; they do not feel the need to understand the exact meaning of it all. I have also seen that some people feel excluded because they don't know what the meaning of the code is. They believe we're a closed and select group of nerds where they don't belong. It is here where I think we have to work harder.

linalab.com

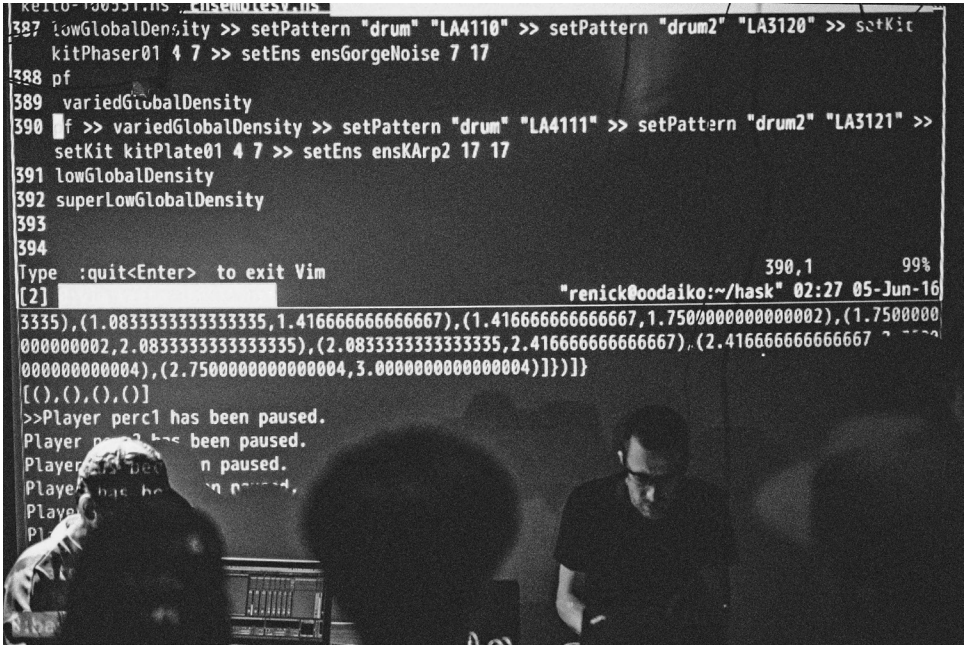


Photo by Jun Yokoyama

Photo by Jun Yokoyama

Renick Bell
US/Japan

I perform with my Conductive system, a Haskell library for live coding autonomous processes and generating patterns of things like rhythm, samples, or event density. I use it to create large volumes of patterns and audition them for audiences; the sounds are new for them and me. I am beginning to use it to control hardware synths and interact with other people's systems.

I have been making electronic music for over twenty-five years. Around 2003, frustration that I couldn't perform it live like I had in punk bands led me to return to computer music languages from university, like Csound and SuperCollider. In graduate school I developed a generative music system with a graphic interface that worked technically but was unsatisfying to play with a mouse. Thinking the limitations came from the graphical user interface (GUI) library, I sought better tool kits in different languages. I had read Alex McLean's "Hacking Perl in Nightclubs" in 2004; it had seemed novel and hilarious, but I didn't consider trying it. However, by 2007 I realized I was making music with code almost as well as I had with the GUI, and the potential became clear: using my typing skills with the enormous powers of abstraction available through live coding, I could go far beyond. I had been watching McLean blogs about Tidal development and wanted to use it, but as it wasn't publicly available then, I started coding Conductive.

As Conductive became familiar, I imagined how the possibilities could be increased. Simultaneously, the software sometimes demonstrated possibilities I hadn't imagined, changing my direction for Conductive. This has given me an inexhaustible list of things to try in performance and features in Conductive to add or improve.

I perform often in Japan, where people increasingly know about live coding or practice it. Of several algoraves we have held in Japan, there have been two in Tokyo where people had to be turned away at the door. I also have many chances to perform internationally outside of the live coding community and observe an increasing interest in live coding.

I see three challenges for live coding based on comments from outside of the community. First, some perceive a lack of immediacy, believing that live coding is not developing as quickly or as much as other electronic music, live or DJ. Second, some feel that live coding emphasizes process to the detriment of sonic results, neglecting aesthetic aspects like the mix of voices in arrangements and other factors related to current developments in various subgenres. Third, recognizing that live coding makes possible things that are hard or impossible for traditional tools, some say that live coding misses opportunities to explore new types of music and express frustration over live coding that incompletely imitates existing genres rather than further exploring the unique territory live coders can access. While we can argue with these perceptions, taking them seriously can be useful. Live coding can progress both technically and in how it is perceived in the electronic music world by accepting these challenges.

renickbell.net

Photo by Ashlae Blum(e)

Ashlae Blum(e)

As a musician with a background in live musical performance as well as engineering physics, live coding is a natural extension of my musical expression and artistic practice. For me it feels almost the same to create music with live coding as it does with an instrument, minus the tactile feel of playing the actual instrument itself and plus a lot more prep work beforehand. I like to experiment with a variety of tools and workflows, and my most fluent languages are currently FoxDot, SuperCollider, TidalCycles, and Hydra, in that order.

As a trans nonbinary musician, I've experienced lots of "othering" in the music industry. After hitting the glass ceiling there a few too many times, I began to question my identity as a musician and wanted to redefine the role music had in my life. I decided that if I could still have an emotional relationship with music, even if I was "just" programming it, I was actually meant to be a musician, and here we are three years later. I started live coding by myself. I just kind of accidentally learned about it on the internet while I was teaching myself Python and, after doing it for a year, was shocked to find that there were actually many people who also live code and that the community was so international. I felt like I'd discovered a kind of lost Atlantis that I had never experienced in other music scenes I'd previously been part of.

One thing I've been working on is a methodology for live coding that uses a nonlinear, layered approach to performance inspired by the modular impressions I've arrived at through making music with live coding software. It draws on aspects of live coding, music production and performance, and methods from physics. I've termed the process *Modular Geometricity*.

My system is an approach or way of thinking about the creative process rather than a specific set of tools. (I guess the tools are the mind.) Since that which we create is but a reflection of our thoughts and the actual software/hardware we are working with (i.e., other people's minds), it is inherently a collaborative effort, even if one is unaware of the history of technology.

All rivers flow to the sea; I think that since most live coding software stems from principles that are deeply embedded in the history of music and technology, fundamentally, most of the current live coding systems are just iterations of the same process. Most of this relates more to communications theory and electrical engineering, I think, than to music itself.

How people see live coding is dependent upon the tech literacy of the audience, which is also inherently dependent upon factors such as economic mobility and demographic representation. Currently, it is an underground scene, which will likely shift if pop and mass media get their hands on it. Generally, I think the idea of coding as art is still quite foreign or even to-be-avoided by the masses. This is okay because it will change.

I see the future of live coding through shader-patterned glasses.

ashlaeblume.bandcamp.com

Photo by Udo Siegfried

Alexandra Cardenas
Colombia/Germany

Live coding is a technique to create live generative art. As a composer, I was immediately attracted to it because it offers the capability to think and create in a way that is close to my own musical thought. From the beginning I was highly interested in music, mathematics, live electronics, real-time compositions, electroacoustic music, and electronic music. At the same time, my interests let me discover different traditional kinds of music from around the world. Through the study of African, Caribbean, and academic percussion, I got deeply interested in the patterning of sounds to create different kinds of music. Through my studies in electroacoustic music and orchestration, I studied with a passion different ways to create and mix sounds in order to create music.

After many years of working with Max/MSP, I found SuperCollider. Working with code offered me the opportunity to get rid of any graphic metaphor and allowed me to describe my musical ideas with a language much closer to what I had in mind—something different than digital boxes and cables and even different than expensive physical electronic instruments. Feeling so comfortable describing music in this way, I spent a couple of years learning the basics of SuperCollider. Even though I was passionate about it, it was a complicated language for me to learn. Thanks to SuperCollider I discovered live coding, and with it I could fulfill my dream of being able to generate infinite sounds, forms, and textures in a live manner. I always wanted my music to sound different every time it was played.

Since the beginning, I have been annoyed by the linearity of time and the linearity of music. I feel I am still beginning on this journey of discovering. I will always feel like a rookie when it comes to creating live coded music. And I love this feeling because I know live coding offers infinite opportunities for music, art, and our society. As a touring live coder, I have traveled around the world teaching and performing my music, helping to exchange knowledge and create communities around live coding, and founding several TOPLAP nodes.

http://tiemposdelruido.toplap.org/

Photo by Antonio Roberts

Lucy Cheesman
SONA, UK

Hi, I'm Lucy, and I'm a live coder.

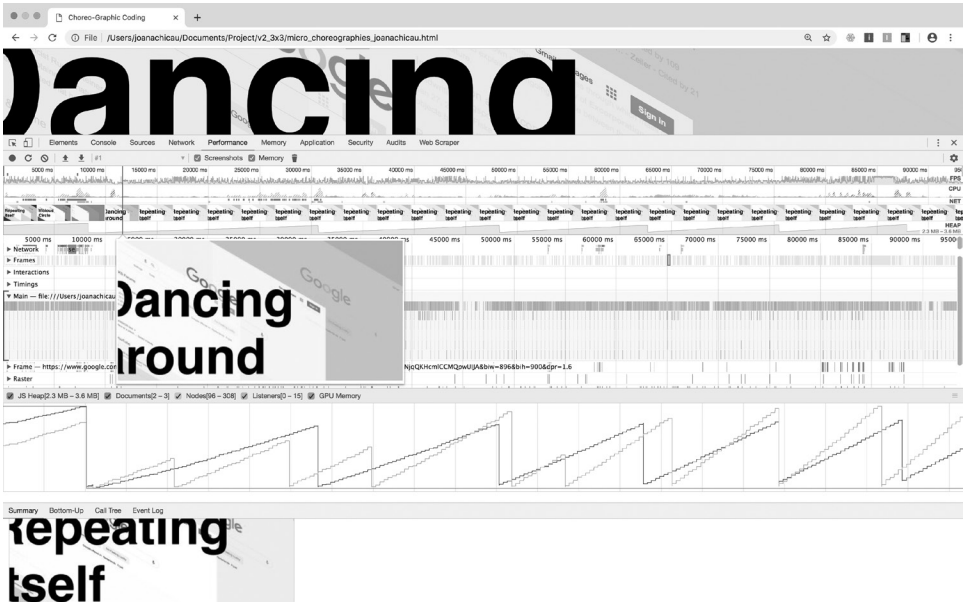
I've been making solo music as Heavy Lifting using TidalCycles since 2016. I still kind of think of myself as a newbie even though four years (and counting) is a pretty long time. I think in part it's because I feel there is always so much more to learn and do. I teach a lot of workshops on Tidal, but I feel like I'm learning as much as the participants every time; my relationship with the computer changes each time we ask a question.

I use various languages, depending on what I'm doing (I've done performances with Tidal, FoxDot, SuperCollider, ixi lang, and Orca), but I'm always drawn back to Tidal, probably because of how it thinks about time and the seemingly infinite possibilities it gives you for sample manipulation. I've always been interested in sampling as a means of making music, particularly taking sounds that are familiar and reusing them in perhaps unfamiliar contexts; Tidal lets me do this in powerful and unexpected ways.

Sometimes my performances are serious or maybe explore some kind of lofty themes, but more often I'm just laughing at my own jokes. At the end of 2015, I went to a live coding workshop for women run by Joanne Armitage and Shelly Knotts, who told us to embrace error; I took that to heart. I think by removing the pressure to be accurate, live coding creates a pretty unique environment for experimentation. In my case I use that freedom to be quite silly; in the picture I'm performing a set using only samples from that year's Eurovision Song Contest (it was happening on the same night), accompanied by some beautiful visuals by Antonio Roberts. I think that writing code can be seen as a superserious dark art, and I hope that irreverent and fun live coding performances can go some way to dispelling that myth.

I'm lucky enough to live in Sheffield, where there is a really strong live coding scene fostered by Alex McLean (a.k.a. yaxu); we have regular algoraves and the wonderful AlgoMech festival. I try to do my bit by running gigs, meetups, and workshops. I was given so much support starting out in live coding, and I really want to pass that on to other people. One of the key features of live coding to me is that openness, and I'm not just talking about free software but the intention to collaborate in nonhierarchical and noncompetitive ways. The next step is to dismantle capitalism—see you there <3.

heavy-lifting.org



Screenshot of Joana Chicau live coding visuals in a web browser

Photo by Marco de Swart

Joana Chicau
The Netherlands/UK/Portugal

In 2016 I started a transdisciplinary research project that interweaves web-programming tools and environments with performance and choreographic practices. Since then I have been investigating diverse notation systems from both dance and web programming, which I then merge into a hybrid form of algorithmic composition: a "choreo-graphic-code" that brings new meaning and produces a new imaginary around the act of coding.

Choreography in the context of this research project is seen as a writing or a metalinguistic space for thinking movement and countermovement in the (de)construction of web tools and digital media environments. The liveness of code writing became a way to activate the choreographic-code, exposing various processes and dynamics of web computing while enhancing the physicality of the body—the body that is in constant friction between the constative (reality describing) and performative (reality producing). Engaging with different forms of choreographic thinking has been a way to bridge and enhance the somatic and semantic within coding.

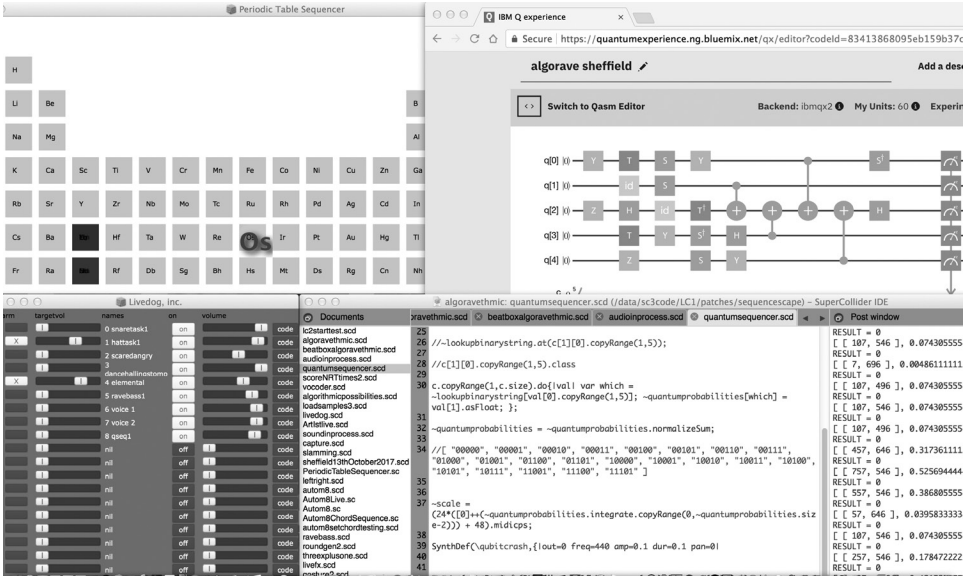
I believe improvisation is key in live coding practices in general, and within my performances I hope to take this notion both as a technical and physical condition and bind the procedural with the conceptual and corporeal layers of live coding.

My practice reflects on the intersection of the body with the constructed, designed, programmed environment. It aims at creating new alternative circuits within the technological sphere of programming languages and possible encounters with the sensorimotor structures that regulate our bodies and movements, hopefully contributing to a different mode of embodying live coding tools and systems of notation and of appropriating digital technologies.

Kate Sicchio has been one of my most important sources of inspiration. A few years ago while I was still writing my thesis, I was introduced to Kate by Alex McLean. Later we all met in person at the ICLC 2016! I still try to follow her work as closely as possible. Another important aspect of my practice is the use of coding languages and its performative instance to build diverse, inclusive, and plural discourses, as well as a site for nurturing collaborative and open work. Such critical thinking and activism on gender equality and inclusion have been brought to discussion within the community by Shelly Knotts and Joanne Armitage, among other feminist live coders. I feel very close to their thinking and have already engaged in public discussions with them.

As in most live coding practices, my system and methodology focus on the use of free/libre open-source models and are concerned with widening the ways in which digital media and computation are presented and made accessible to the public.

joanachicau.com
jobcb.github.io



Periodic table sequencer and IBM six qubit frontend in SuperCollider. Image credit: Nick Collins (Live Dog, Inc.)

Nick Collins
Durham University, UK
composerprogrammer.com

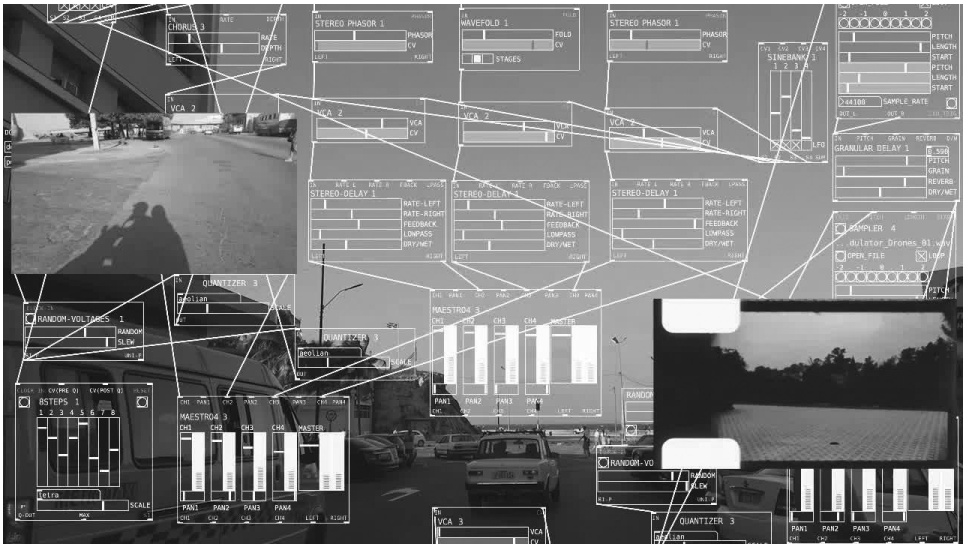
Jack Code's Rebellion

A single character change in text can have enormous consequences. In Terry Gilliam's administrative nightmare *Trazil* (1985), *T* becomes *B*, *Tuttle* becomes *Buttle*, exchanging life and death; in the page title, Jack Cade's fifteenth-century rebellion takes on a contemporary repercussion. From Shakespeare's *Henry VI*, part 2, subtly rewritten, we now find a beautiful anticipation of the theater of live code: "These hands are free from guiltless blood-shedding, This breast from harbouring foul deceitful thoughts. 0, let me live! Code: I feel remorse in myself with his words but I'll bridle it; he shall die, an it be but for 'pleading so well for his life . . .'"[3]

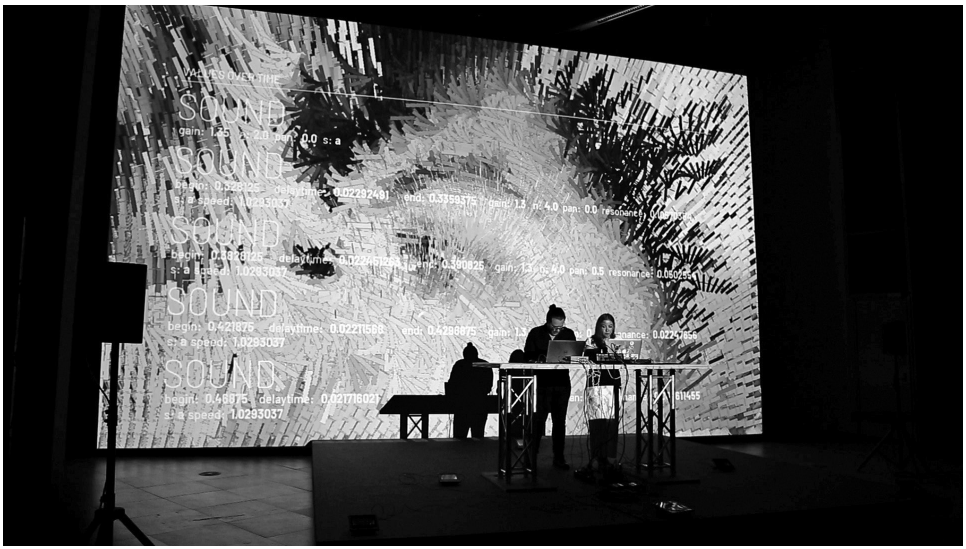
Changing your mind while you act in the world is nothing new; it is at the heart of survival and improvisation, an essential part of the history of humankind, with language's versatile generativity a great creative backdrop. But explicit interaction via a computer programming language, as artistic activity often framed on a concert stage, was a step change in intent of which I have been privileged to be some small cog.

The historical precedents are fascinating, from mathematics and games, through role playing to drama, touching on political systems and the cultural history of algorithms. Varied topics from human life can underlie live coding dramaturgy, and this wider net of human subjects has provided a rich grounding for much of my own live coding work. In the algorithmic choreography laptop duo Wrongheaded (with Matthew Yee-King), once-off performances included a live teddy bear dissection with liberal tomato sauce, a séance evoking the ghost of Alan Turing, and "The Gospel according to Wrongheaded."[4] A more recent and I hope never fully formed project, Live Dog, Inc., has drawn on varied inspirations to motivate improvisational coding, from satire of existing music (Johnny Bash's "Live from Fortran Prison," HadIORead's "Discothick") to hard science including a live-code-controlled periodic table and six-qubit quantum computer experiment sequencers.

I wonder if what I think about live coding has changed over the years; that would seem an entirely appropriate state of affairs. I revisit my attitude to live coding once in a while, though I wouldn't like to reinvent the repeal. Yet re:rereading some "thoughts on live coding" I wrote quickly in 2004,[5] I find that I wouldn't change the text so much, and the profoundly deep live coding insight I crave remains beautifully elusive. Perhaps I should leave behind the occupation of live coder, for as a bezzific proverbial 1857 poem cries, "A change is as good as a zest,"[6] where *T* is to *B* as *R* is to *Z*. As Click Nilson once wrote, " ".[7]



Malitzin Cortes, "Generative promenade or Nightmare catcher," Modern Art Museum, Mexico City, 2019.
Photo by Malitzin Cortes



ACI Asia Culture Center. Foodhack 2019 ISEA Korea 2019
Photo by Asia Culture Center

Malitzin Cortes (CNDSD)
TOPLAP Mexico

Live Cinema Coding
Nonlinear stories, abstraction, and audiovisual synesthesia in the contemporary context of code

In the 1960s, artists and filmmakers began to challenge the conventions of audiovisual language, creating more participatory roles for the spectator. The constant development of design, film, and computational technologies pushed these works further forward. They left linear, classic cinematographic language behind, taking film outside cinemas and into art galleries, abandoned factories, and outdoors, implementing different forms of experimentation through multiscreen projections. A great reference for what we can see today, which in 2020 is seen from a distance with our festivals of electronic music and digital art and the "language of new media,"[8] is *Light Music* (1975) by Lis Rhodes. It is composed of two "films" projected in a smoky room, with an intense sound composition created from flashing patterns on the screen, demonstrating that emotional, perceptual, and immersive experience is nothing new.

Live cinema coding is a hybridization between live coding and more established practices of expanded cinema and live cinema, generating new stories and open interpretation. These practices have their own tools, with possibilities that are not only technological but also aesthetic. On-the-fly programming creates sound phenomena conducive to these new possibilities and is able to generate music of any genre. Visually, it has joined the world of generative creativity and computational thinking in the arts (creative coding) and is available to all artists, not only those specialized in science and technology.

The creative work and research that I have carried out with digital artist and programmer Iván Abreu over several years has combined our technical skills and artistic passion for the console, not only as a way to preproduce audiovisual content, from the keyboard and logic of a program, but also to tell stories and compose music that emerges from, and through, algorithms. In this way we have used different tools for live coding, such as SuperCollider, Hydra, Orca, and Pure Data, but above all, for its efficiency, community, and functional way of creating and ordering anything in time, we use TidalCycles. It can generate complexity from very little code, and we have found enormous poetry in the syntax that takes us from within to achieve a sound and visual idea. From the mind to the algorithm and then to the screen, a palindromic pattern can be sonic and/or visual, evoked by typing Tidal's palindrome function from the keyboard. We are now developing and contributing systems back to the community, not only to understand the complexity and compositional beauty of Tidal but also to find ways to hybridize different tools and programming languages with a single reason: to tell intense stories with sound, light, and images in real time.

malitzincortes.net
https://vimeo.com/cndsd

Photo by Mamady Diarra

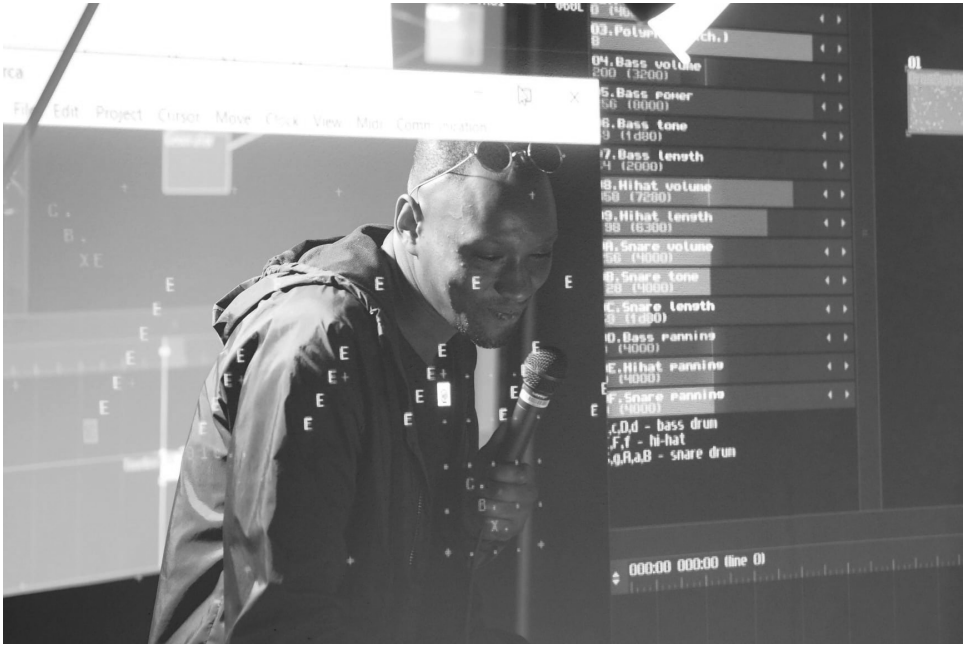


Photo by Mamady Diarra

Mamady Diarra
France

Live coding? For me this is really brand new. I had already learned some programming, nothing more. . . . I discovered live coding while watching an interview with someone talking about making music through code with a system called Opusmodus. The price of this software calmed my interest, but I was curious. Then along the way, I started to find out more. A friend told me about a live coding workshop that would take place in Paris organized by NO School Nevers! I went for it without hesitation. I was excited to meet the creator of the open-source software TidalCycles (Alex McLean), which I use most often in live coding. The coding aspect, the accessibility, and even the active, caring community led me to use this software over any other.

I'm always excited to attend an event that brings together other live coders. It's always informative and really interesting. My first live coding gig was at the 7° Festival Ambient de Paris in September 2020, where people gathered outside of the live coding scene with people who had mostly never heard of live coding. I was surprised. The audience was receptive and came to ask me questions at the end. It's heartwarming to see how live coding can affect so many people.

The live coding system is an experimental field, constantly questioning itself. This is unique to information technology and to tech in general. But it is true that in the live coding system there is a lot more unpredictability. Sometimes I feel like I spend more time engineering . . . solving problems . . . It's alive!

https://afalfl.bandcamp.com/

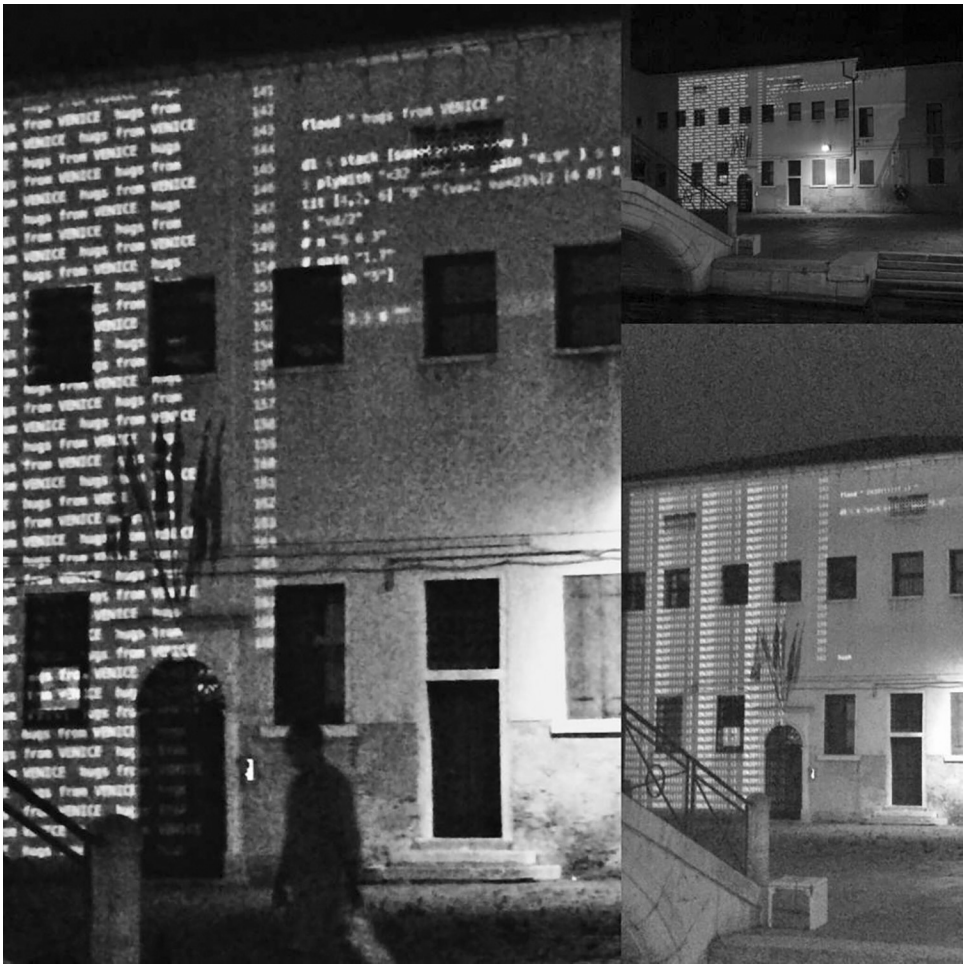


Photo by Claudio Donaggio

Claudio Donaggio
TOPLAP Italia

The year 2015 was a difficult one for me, both physically and mentally. One day that year I saw a video on YouTube by Andrew Sorensen. At the time, I had never written a line of code in my life but I was fascinated. I started immediately with Sonic Pi. Live coding and, even more, the community of live coders around the globe helped me to gain enough self-confidence to beat my own demons and believe in myself. And that's how I started. To me, live coding means "complete freedom from genre and labels," which is what I have always been looking for in musical and artistic terms. Being able to talk with the machine through code and improvise electronic music is the core essence of live coding practice, in my opinion.

I like to work with different environments and languages. I would not know where to start in order to create a system like Tidal or Sonic Pi, but I try to be as much of a polyglot live coder as possible! I am one of the administrators of the Live Coding Italia community on Facebook and Telegram. I hope one day that my contribution will be to have spread the practice around as much as possible in my own country.

My hope is that live coding will not only be designated to the academic world but be adopted, as a technique, by more performers and artists. Nothing against it, but "Be wary of institutions" is my motto. I do think more and more people will become interested in the possibilities of live coding, and I try to help newcomers as much as the community helped me when I started out.

I've seen different reactions from people to live coding. At first they are like "This is something for nerds!" and then when they experience an algorave, they're gobsmacked! Generally, I think it is perceived as difficult to understand, initially. That's why it is always good to start a performance event with a talk, in my own opinion and experience.

https://twitter.com/Etol_livecoding

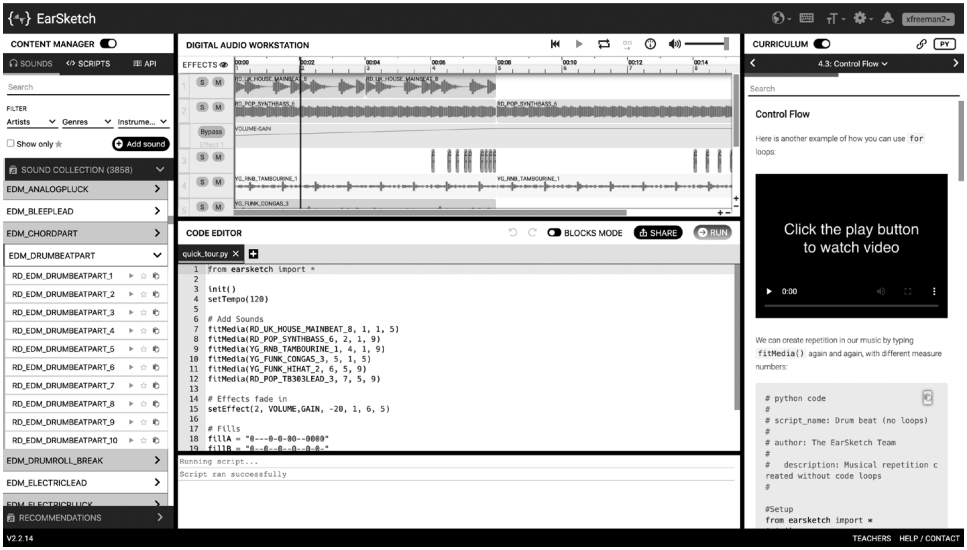


Image credit: Georgia Institute of Technology

Jason Freeman
Georgia Institute of Technology, US

As both an artist and a researcher, I use technology to engage others in creative and collaborative musical activities. While I do occasionally perform myself and create music for professional musicians to perform, I far prefer to devise contexts in which anyone can express themselves musically.

I was initially drawn to live coding as a performer (in the early 2000s) because I could reveal the often arcane practice of algorithmic composition to an audience in a high-risk, improvisatory way. In those early performances, audiences were impressed by the complexity of what I was doing, but they did not understand it at all. I found this dispiriting and abandoned live coding altogether for a few years. Beginning in the late 2000s, I found new ways to approach live coding that were more in sync with my own sensibilities. I developed new environments that shared the experience of musical coding with others instead of sharing my screen with an audience.

Live coding can be a virtuosic activity that is the culmination of years of practice. It can also be a test kitchen that pushes the design boundaries of computer music languages. My practice is neither virtuosic nor innovative in these ways. Instead, I explore how simple systems that are grounded in familiar paradigms can get novices excited about creative coding.

My students and colleagues and I have created a couple of environments over the years, but my current focus is EarSketch (created with Brian Magerko and many others). It is designed for kids who don't know much about music or coding. You write code in Python or JavaScript to remix loops from an audio library, and it shows the results of executing your code in a multitrack audio view. It very quickly gets kids excited about music, and coding, and sharing what they've done with their friends (as the photo illustrates). While it does support live coding, it's the coding (not the liveness) that is the focus.

I admire the many virtuosic live coders out there, but I'm not one myself. I don't perform often as a live coder anymore and when I do, it is usually in an educational rather than a pure performance context. For me, the classroom is the most exciting place for code to be right now, whether live or not.

EarSketch has a much lower barrier of entry (both musically and computationally) but also a much lower ceiling than other systems. It's also deeply situated in commercial music—both the library of audio loops it uses and the DAW paradigm that inspires both the interface and the programming API.

http://earsketch.gatech.edu

Photo by Federico Sande Novo. Courtesy of Museo de Arte Moderno de Buenos Aires.

Photo by Federico Sande Novo. Courtesy of Museo de Arte Moderno de

Photo by Nahuel Zeta

Flor de Fuego

Facultad de Artes, Universidad de La Plata, Buenos Aires, Argentina

Nowadays, live coding is one of the most important practices for me. I explore and try to hybridize sound/poetry/visual/body in my live coding performances. I was prompted to engage in this research from wanting to create audio/visual sets and then found a whole world of limitless possibilities.

I'm an active collaborator within the Hydra community (software made by Olivia Jack) and help organize the Hydra meetup. I give a lot of talks and workshops around this software and also tend to explore and research it. I teach Hydra in high school for teenagers, and I'm also an active member of the CLIC (Live Coders Collective/Colectivo de Live Coders). I think my contribution to live coding practices is related to education but also art because I have an art university education background, and I'm really influenced by that.

I do live coding alone, but I'm also part of collaborative projects, such as c0d3 p03try (Rapo and Flor de Fuego), and lately have been researching potential connections between net.art and Hydra, together with Naoto Hieda. I also participate in different algoraves from mostly Latin American communities.

I imagine live coding will further develop, with more collaborative interactions and hybrid performances and with many more people getting involved. Currently, I think people usually don't understand what live coding is until they see a live performance, and when that happens they seem very curious about it. Is not enough to explain it; you have to experience a live show.

https://flordefuego.github.io/

Image credit: Molly Gunn and Westley Hennigh-Palermo

Sarah Groff Hennigh-Palermo
Codie, US

I started live coding via Live Code NYC meetings in 2017. The meetings were essentially an excuse to see friends and yell about compilers. The "actually performing" part I was less sure about. Then Kate Sicchio convinced me that if I performed with her, I would get to write a visuals framework. Being the kind of girl who is unable to turn down the chance to write an SVG (scalable vector graphics) framework in ClojureScript, I agreed. We performed once, I fell in love, and our collective, Codie, was born. The third member, Melody Loveless, joined us a few months later.

When we started, the New York scene was still relatively new. Kate had moved to New York from Sheffield and founded the group with a few other folks. This was a great opportunity for us because the group was small enough to allow everyone to play every show they wanted, and so in 2018, we played something like twenty-five or thirty shows. These were mostly in dark bars in Bushwick, but we also did some more reputable venues.

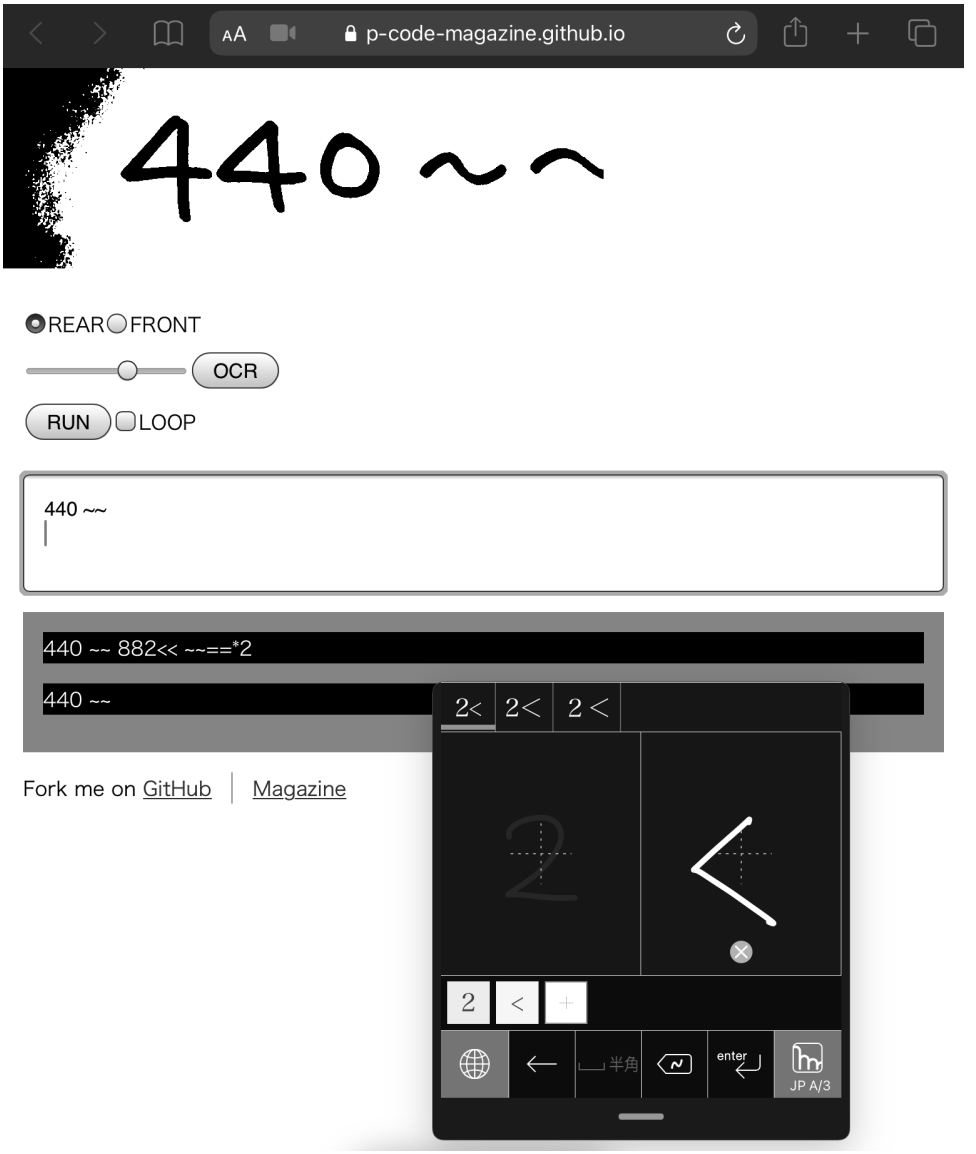
I think the New York scene then was, for better and worse, a bit "artier" than in some other places—people understand our shows as art events and expect a lot more weirdness than bangers, per se. It also meant that there was a lot of focus on pairing musicians and visualists before shows and sometimes even practicing together. Codie takes this to an extreme in that we practice a lot; we're a proper collective. I like to think we were an example for the rest of the scene.

This approach—rather than visualists showing up the day of and asking who wants vis—is really vital because it opens a space for deep collaboration. Instead of linking together sound and light through audioreactive visuals, which I always find a bit dull, we are able to link through a basic agreed rhythm and then play off one another in a way that's only really possible when you play together a lot. It makes for a really rich dialogue. We've also elaborated on it in short films we've done together.

This way of computing also helps me "unthink" the engineering I do as my day job. It allows for a relationship with computers where they are more like plants, rewarding cultivation and experimentation. That experience then flows into my nonperformance art, which has become a counterpractice to the sort of technical mystification and dystopic AI projects that are popular now. I'm looking to unfold a space outside new media hucksterism and unquestioning engineering positivism.

In the future I would like to see more attention paid to the seriousness of live vis. It still feels like we are the little sisters in a lot of cases. There are not always vis talks at conferences, and at shows you often get the setup equivalent of playing your tunes through a boom box. But it's not like I'm organizing anything these days, so maybe I should step up instead of just wishing!

http://art.sarahghp.com/



Screenshot by Akihiro Kubota

HAUS++
(Hirozumi Takeda, Yosuke Hayashi, Takanobu Inafuku + Akihiro Kubota), Japan

P-Code is a minimal language for live coding sound performance.[9] The code is interpreted from left to right, divided into numbers and other symbols, and then executed. All symbols that cannot be interpreted are treated as white noise, making it an error-free language.

The syntax of P-Code is minimal, but the input methods are diverse. You can write code in a variety of ways, including via keyboard, image optical character recognition, handwritten input, and voice input.

Collective live coding sessions with multiple participants are also possible. P-Code Playground is a chat environment where the P-Code can be executed. The codes and texts entered by the participants are recorded, along with the time they are entered, and can be replayed using the playback function.

R3PL (random regular expression read-eval-print loop), the new runtime environment for P-Code, now allows input with regular expressions. Code written in regular expressions is, so to speak, a potential code that is fraught with indeterminacy.

Live coding is a never-ending open challenge to go beyond "improvisation."

p-code-magazine.github.io

```
do
  let rfreq = segment 1
      seq = "{1@7 1@11 1@21 1 1 1 1@3 1@3 1@3 1}%16"
      dg = reduce
      chain mod p = sb 0.2 (mod) $ one 0.3 $ shift $ one 0.2 $ shift $ roll 0.2
                    $ dg $ struct seq $ p
  d1
    $ cpsDisc minTempo maxTempo
    $ limit
    $ stack [
      padOn $ slow 3
        $ (|+ note (shiftBy 4 $ choose [0,12]))
        $ (|+ note (scale "ritusen" "{0 -2 -4 -1 -3}%16")) $ pad5n
      , samplestart (segment 1 $ range 0 127 $ slow 10 rand) # midichan 4
      , sampleend (segment 1.1 $ range 0 127 $ slow 21 rand) # midichan 4
      , useIter (iter 16) $ stack [
        kickOn $ chain (# pad4n) $ pad3n
          # legato (range 0.5 1 $ shrand 100)
        , harmorOn $ (2 ~>) $ chain (# gain 0.4) $ harmor # note "-12" # speed007chan
        , perc1On $ often (off (0.0625/2) id) $ chain id $ pad11n
      ]
      , perf4 (segment 1 $ range 0 127 $ shrand 101)
      , perf7 (segment 1.3 $ range 0 127 $ shrand 103)
      , perf8 (segment 1.25 $ range 0 127 $ shrand 202)
      , scene (segment (1/2) $ range 0 sceneRange $ shrand 9991)
      -- harmor cc's
      , stack [
        ccv (rfreq $ ccrand 10 100 133) # filter1
        , ccv (rfreq $ ccrand 10 100 233) # filter2
        , ccv (rfreq $ ccrand 0 127 333) # unidetune
        , ccv (rfreq $ ccrand 0 64 433) # phwidth
        , ccv (rfreq $ ccrand 40 80 38282) # phspeed
        , ccv (rfreq $ ccrand 5 127 533) # envattack
        , ccv (rfreq $ ccrand 0 127 7283) # envdecay
        , ccv (rfreq $ ccrand 100 127 633 ) # filter1width
        , ccv (rfreq $ ccrand 127 127 733 ) # filter1env
        , ccv (rfreq $ ccrand 100 127 833 ) # filter2width
        , ccv (rfreq $ ccrand 127 127 933 ) # filter2env
        , ccv (rfreq $ ccrand 0 127 1033) # timbre
        , ccv (rfreq $ ccrand 64 64 1133) # amprel
        , ccv (rfreq $ ccrand 0 127 7) # harm
      ] # speed007chan
    ] # cps (140/120/2)

    hush
```

TidalCycles code.
Image credit: Mike Hodnick

Mike Hodnick

When I first saw TidalCycles, I immediately recognized that it excelled at rhythms and patterns. Being a programmer and a percussionist, it looked like the perfect match for me. But I saw further than that; I envisioned that TidalCycles could create endless permutations of complex rhythms spiraling out of control. I was also tired of making music with a DAW because it was tedious. TidalCycles helps automate the music I really want to make.

TidalCycles allows you to write computer code to make music, but not just any music; it allows you to bend and break dozens of patterning and sequencing conventions. Through small amounts of code, you can create minutes of ever-changing musical patterns. I think it's arguably one of the greatest music sequencers ever made.

It's a cliche, but the best musical moments are when accidents happen. TidalCycles exponentially increases the frequency of those accidents not only because it is live code but also because it requires very little code. Small changes can result in large musical differences. I've built my musical practice around experimental workflows that exploit specific features or quirks about TidalCycles; for example, "What if I create ten patterns that focus on TidalCycles's *XYZ* this week?" Through Tidal's MIDI capabilities, I've also been able to take these experiments to hardware and software synths, which has really opened up the sonic possibilities.

I started out with the 365TidalPatterns project, as my way of learning Tidal out in public. The goal was to create a small one-minute pattern every day with the samples I had in my own library. I never really set out to make those patterns into a final, cohesive project. I valued quantity over quality. The day-to-day flow made me comfortable with doing small experiments and shelving them. I had source code and an audio file for every pattern. I learned much later that I could then go back to those patterns I liked best and shape them into final compositions. I continue that practice to this day.

The international live coding community is one of the most open and welcoming groups of people I have ever known. It is continuously helpful, and we all support each other's efforts to learn and create. The community also actively works for positive changes in diversity in the digital arts. Helping build a local community has been challenging. It takes a lot of persistence!

http://kindohm.com/

Photo by Zuzanne Zgierska

Photo by Paulus van Dorsten

Timo Hoogland
nl_cl (Netherlands Coding Live), HKU University of the Arts, Utrecht

Before I started to live code electronic music, I was mostly spending my time performing with drums and programming audiovisual installations and compositions. I've always felt the urge to perform onstage, so I was looking for ways to translate my programmed audiovisual works to a live performance. By connecting a controller to the software, I felt disconnected from the music and sound when only controlling dials, sliders, and presets. While with drumming I can improvise, be in the moment, get into a flow, and have this immediate connection between my movements and the sound, with the controller and computer I felt restrictions that did not improve my creativity onstage.

My first encounter with live coding was an eye-opener in the sense of how creative coding and electronic music could be performed. Live coding is a form of expression that most closely matches my performing style with the drums. Via coding I can combine and explore many fields of interest, such as polyrhythms and algorithmic composition, in real time. Also, similar to the drums, I can extend my instrument with new functionalities, adding new flavors to my sound palette and either rehearsing performances or improvising onstage. On top of that I can also let the code give me new and unexpected results, as if my instrument is also partially the composer or band member.

I experimented with systems like Max/MSP, Sonic Pi, and SuperCollider but decided to design my own language that fits my approach to performing electronic music. The result is Mercury, a language focusing on the quick expression and communication of live coded audiovisual performances. The live coding scene strives for openness, transparency, and inclusiveness, and for that reason Mercury code has clear and human-readable syntax. This allows the performer to focus on the composition and keep the audience engaged. The responsive text editor adjusts font size to keep code visible, similar to the earlier Fluxus live coding environment. Furthermore, the editor only allows thirty lines of code. This restriction forces me to erase code and make creative choices during a performance.

In my latest performance, I combine drumming and live coding, exploring ways to improvise with electronic music while playing an acoustic instrument. The computer functions as a second performer, listening for patterns played on the drums and making decisions to change the code.

Up until now I think the audience for live coding performances could be categorized into two groups. One group would be people familiar with code, who mostly understand its effects on the sound or visuals. The other group consists of people encountering live coding for the first time and who are surprised by its possibilities. Some may even find it impressive and almost like "magic." But as creative coding becomes more widely known and coding is taught in more places, I see that live coding for its own sake starts to have less impact, forcing artists to start searching for new live coding approaches or be more critical of their output. In the (near) future, live coding will surely find its way into a wider variety of genres and disciplines where the focus is not necessarily on the live coding itself but more on the results.

https://www.timohoogland.com

Photo by Antonio Roberts

Miri Kaat
UK

I am a UX (user experience) designer, researcher, and strategist with ten-plus years of professional experience across music, games, and education, and my educational background is in game design and web technologies. Just making impressive technology isn't enough—it needs to have an emotional impact. I work toward bringing digital interfaces to music creation, producing new and accessible ways for artists to interact with technology. Working in the music industry has enabled me to work as an artist, designer, and technologist. It's connected me to a huge network of creative practitioners, industry practices, learning resources, tools, and technologies.

From working in this industry, I have also experienced the gender and racial imbalance. I wish to address this more by teaching, inspiring, and enabling people from minority backgrounds and introducing them to the joys of live coding. I have been involved with live coding as a performer, music technologist, and educator. I have also immersed myself in the algorave music scene. Perhaps most importantly, I've worked to help others do the same, hosting events and workshops for women and nonbinary people, who often feel excluded from music technology.

My artistic practice is in audio-responsive multimedia. I use SuperCollider with TidalCycles for live coding music and Max/MSP for generative audio and visuals. Being a musician, influencing the creation of music technology solutions, and coming from a minority background give me the insight to provide music technology training for underrepresented groups. I am lucky that my interests lie at this intersection in music, technology, and accessibility. This, in turn, feeds my UX design practice.

I am passionate about science, technology, engineering, and mathemathics education and promoting music technology skills for people from minority groups. I have conducted workshops and performances all over the UK and internationally. My aim is to create an impact using the power of music, art, and technology as a force for social change. I am among some of the pioneering women in live coding within the UK. I believe live coding is especially good for teaching creative coding skills. I released my debut EP with Establishment Records in 2017, a release that saw media acclaim from outlets like *Resident Advisor*. This was composed and recorded using SuperCollider and TidalCycles.

I believe the live coding scene is poised to reach a much greater level of diversity, collaboration, and innovation in the future. I believe that artists and musicians can feel disenfranchised with technology; this is a major issue that the music tech industry faces right now. Inspiring and enabling people to work with new music software solutions is a step toward bridging that gap. My long-term goal is to enable and mentor minority leaders in music technology.

https://mirikat.github.io/

Livecoding jam with Abhinay Khoparzi, Akash Sharma, and Joshua Thomas.
Photo by Dhanya Pilo

Workshop on the Marching JS livecoding platform.
Photo by Abhinay Khoparzi

Abhinay Khoparzi
Algorave India

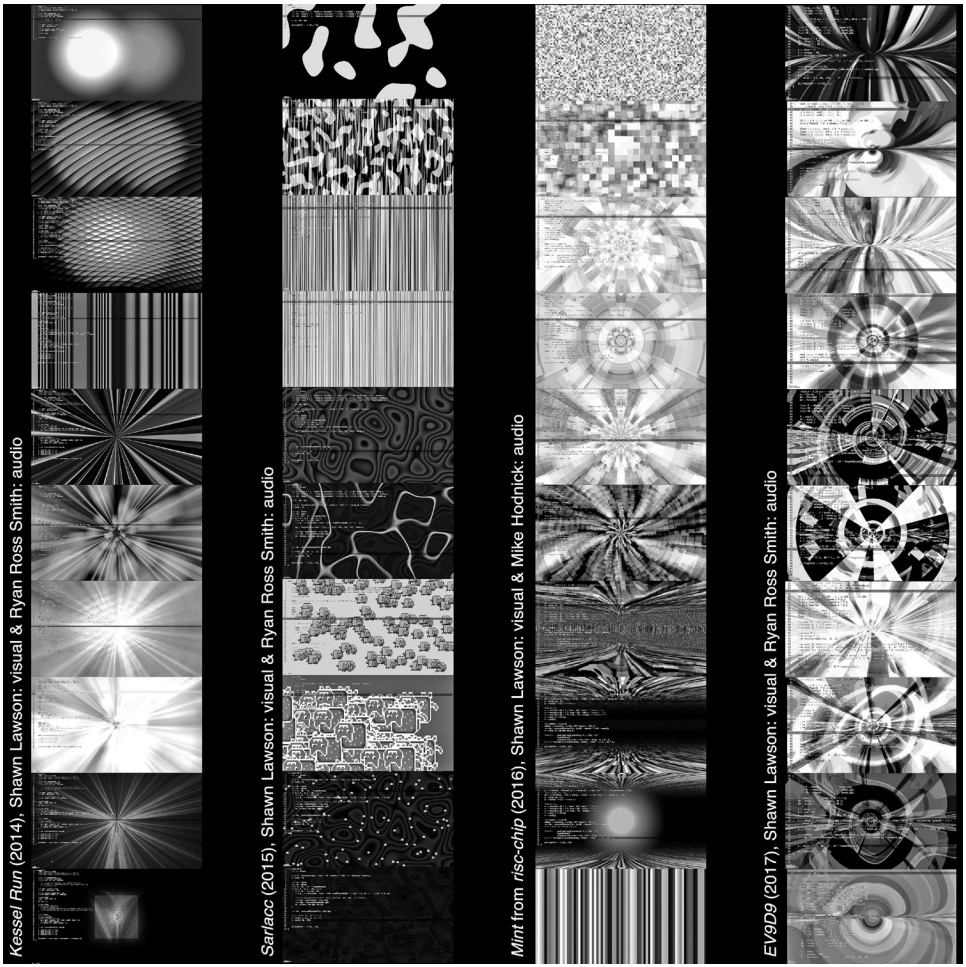
Although there had been workshops and one-off events, a live coding "scene" didn't develop in India until Algorave India came to being in 2018. The first few events at Algorave India were supported by the Open Codes project by the Goethe-Institut and Zentrum für Kunst und Medien (ZKM) as part of their *Open Codes* exhibition. This happened after some friends and I (including longtime practitioners such as Rebecca Fernandes and Akash Sharma, as well as visionaries such as Dhanya Pilo) saw the renewed interest in the experimental/noise music scene brewing in the bigger cities in India. We wanted to recreate a truer form of the experimental gigs we used to organize a decade earlier, as well as showcase local talent that had been lurking in the shadows.

We soon moved to independent events funded by the community and ticket sales, starting with a postworkshop event with a local creative-coding community at Walkin Studios, Bengaluru. These independent events seemed like a better way to engage with and bring together fragmented creative-coding communities in the city. Even though there had been a long-lasting subculture of algorithmic practices in the country, it had always been hidden away in art galleries that mostly highlighted European and US artists. Algorave India and live coding in general have fared well in making the creative-code scene more approachable, reaching outside the realm of "geeks" and art nerds.

The immediacy of results and the quick feedback in the process of live coding have been especially useful in getting people out of initially feeling intimidated by programming and mathematics in general. Performances themselves become miniworkshops where one-on-one and one-to-many interactions with audience members can develop into a learning space. Many contemporary artistic practitioners who were stuck in small towns, including people who didn't believe they could show their beginner code and early experiments, came out to events and shared their works on various forums and social channels.

Engaging with other communities, specifically in Canada and Latin America, has been especially fruitful in terms of collaboration and the exchange of cultural ideas, as well as crossover events that feature artists with different skill levels joining network music ensembles.

https://khoparzi.com
https://algorave.in



Four time-lapse examples of live coded graphics.
Image credit: Shawn Lawson

Shawn Lawson (Obi-Wan Codenobi)
Arizona State University, US

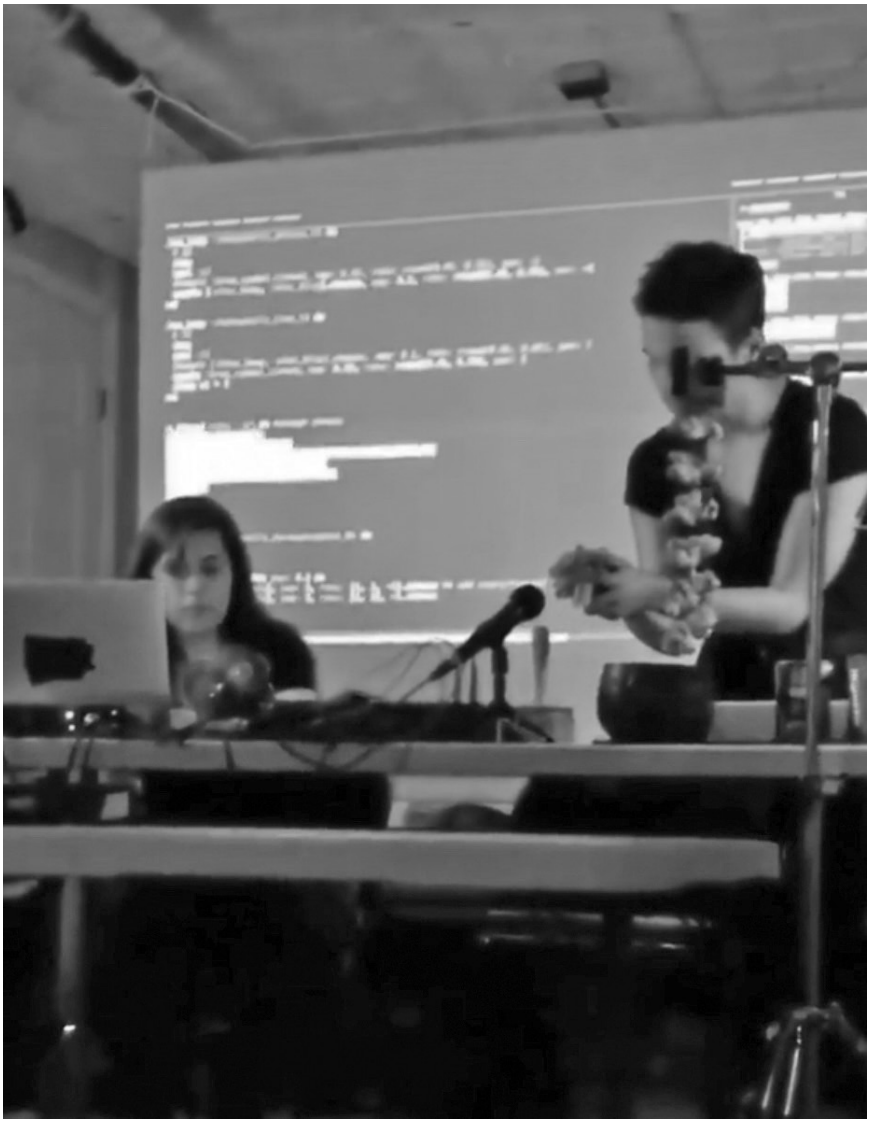
I often describe live coding as driving a race car while simultaneously attempting to change the engine oil: plausible, dangerous, and exciting because the audience is there not only for the roaring success but also for the spectacle and possible tragedy of the crash. Then I continue on by saying they will be able to see the code I'm writing as it's written in addition to the visual performance. In a way, there's a demystification of live performance magic, yet I think this makes the experience more sublime because the viewer is along for the ride (a passenger) in this race car.

I've built a few live coding editors—namely, the Force (GLSL), the Dark Side (GLSL and TidalCycles), and LiveWare (GLSL and Lua). The Force autocompiles and, if successful, then executes the GLSL while you're typing. The Dark Side mixes the TidalCycles language into the Force so that both can be written in the same text buffer by multiple users telematically. Lastly, LiveWare is a Lua binding on top of LibCinder with additional functionality brought in from the Force. Each has different degrees of Open Sound Control (OSC), MIDI, and WebSocket connectivity. All have audio input.

I suppose my live coding practice is heavy on tool building and my own very individualized software needs. With the Force, there were other GLSL live coding systems, but none were designed or tuned for performance. For the Dark Side, my Rebel Scum collaboration with Ryan Ross Smith encountered a physical dislocation. We needed a method of continuing the collaboration telematically without compromising audiovisual quality. With Liveware, I was feeling the need to explore graphics outside of the bounds of GLSL. I'll have a visual idea and then spend time creating an environment that is capable of supporting the vision if it doesn't already exist. Even well before I started live coding, I was writing my own interactive art software. I find that off-the-shelf software is typically too formulaic, constricting, or not quite capable. While this is an extra burden (toolmaking), I think it's accepted as part of the practice.

Media-wise, there are a limited number of live coders for visuals. Online forums for visualists and internet platforms (Vimeo) have been successful at allowing us to find and share progress with each other. With collaborators, we rehearse material, try new material, talk quite a bit about what kind of material we should assemble for the next work, and strategize which calls for proposals to apply to. That all sounds more businesslike than it is in reality, which is significantly more free-form and even chaotic.

http://www.shawnlawson.com



Melody Loveless and Caitlin Cawley.
Photo by Melody Loveless

Melody Loveless
Codie, US

I initially started live coding after being encouraged to join LiveCodeNYC, a meetup group that organizes meetings and algoraves in New York City, by Kate Sicchio and Sarah Groff Hennigh-Palermo. Afterward, Kate and Sarah invited me to join their live coding audiovisual group, Codie. Since then I live code in multiple projects, including a collaboration with percussionist Caitlin Cawley and a solo project in which I record and manipulate samples of my voice.

Objectively, it makes sense that I would be drawn to live coding. I am trained in percussion and music composition, I work with creative and interactive technologies, and I've enjoyed patterns, minimalism, and conceptual art for a long time.

A major quality that drew me to live coding was how processes are highlighted. I often use process as a way to structure events and as a metaphor for ideas that I am meditating on. For example, in my sound installation *Memory Room* I juxtaposed and alternated between field recordings of water and pink noise as a metaphor for how memories can be replaced over time without people noticing. When I understood that live coding highlights processes, I immediately became intrigued.

My initial experiences teaching live coding to kids and young adults with special needs and social-emotional learning disabilities especially inspired me to think deeply about my role as an artist and educator. As is the nature of most creative technologies, live coding has the potential to empower people to participate in and create art—regardless of their artistic knowledge or comprehension of the underlying systems being used. My aforementioned experiences specifically highlighted this idea and showed that reducing participation in music to the execution of syntax has the potential to open performance opportunities to people who would not be able to participate otherwise. For example, the loop-based nature of live coding technologies could allow people who struggle with movement, like those with motor skill disorders, to "play in time."

Pedagogically, live coding has been a great tool for discussing and demonstrating electronic audio and visual principles. For example, when introducing Hydra's operator functions, I also discuss pixels and additive color mixing while demonstrating mathematics in action. I also emphasize the idea that current live coding technologies are a continuation and extension of previous technologies and art movements by contextualizing these tools to their inspirations and related technologies. Examples of this include how I introduce analog video synthesis before demo-ing Hydra and how I connect live coding music to tape music, turntables, synthesizers, and process music.

When performing, the role of time and synchronization varies greatly depending on the project. In Codie, we often just start our technologies to begin a performance. Once performing, Kate and I will refer to Sarah's visuals for cues. In contrast, Caitlin and I discuss frameworks for improvisation ahead of time and sometimes coordinate specific moments. While each collaboration has its own challenges, when performing alone, the added layer of singing makes this project the most difficult to execute.

melodyloveless.com

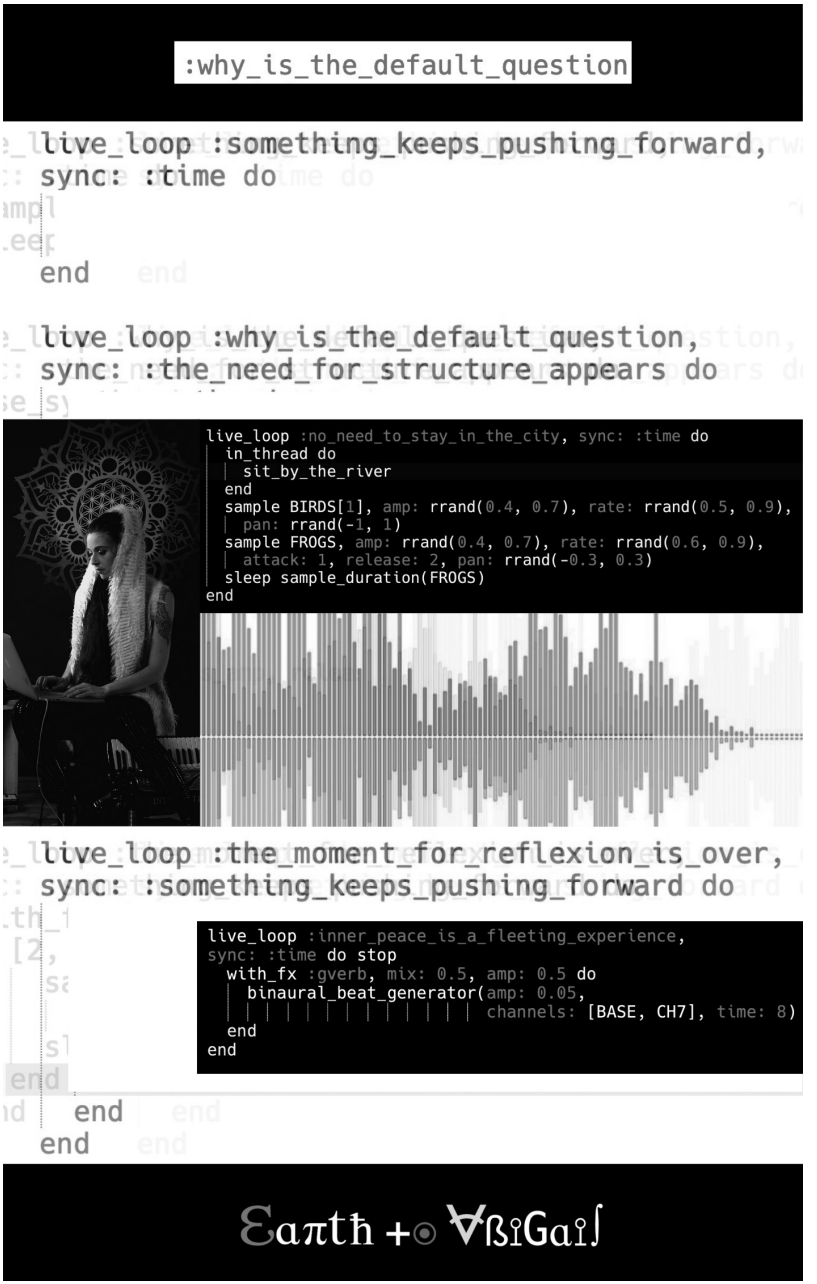


Image credit: Mynah

Mynah Marie, a.k.a. Earth to Abigail
TOPLAP Israel

When I started performing as a live coder, people would ask me, after my shows, "Why choose live coding? Why not use a traditional DAW?" It's a fair question. I remember asking myself this many times at the beginning.

When I discovered Sonic Pi—the live coding platform I use for most of my performances and streams—I knew I had found something incredible. I didn't know exactly why it was incredible; I just instinctively knew it was. I felt the possibilities without having the knowledge to experience them clearly. I became obsessed with finding answers to this initial "why." It awoke a curiosity in me I didn't know I had and kept me up at night with a thirst for knowledge I never seemed able to satiate.

This "why" helped me find my creative freedom. Earth to Abigail isn't me—it's the space I'm creating in. I'm not alone onstage. When I'm live coding, my laptop is more than a tool. It's a companion, a source of inspiration, an entity I can have a conversation with. I'm the master of my own creative universe, and through this conversation with my computer, I have access to constant sources of inspiration, challenges, and surprises. Ultimately, it's this conversation happening through the code that keeps the creative process interesting and evolving. Because of that, you could say it's a love for communication that drives my passion for live coding—communication with an artificial "being" that has capacities I don't possess and communication with the audience through the music and the words on the screen.

Programming languages have an expressive value beyond their initial function of "building things." And computer languages are mostly rooted in the English language. Music is also a language, with its own set of rules, vocabulary, and syntax. When I'm live coding, most of the time what I'm really doing is "painting with sounds" the story I'm writing with words in the code projected on the screen.

Now the initial question of "why" has been replaced by "why not?" Why not strip away all the boundaries of why people write code in the first place? Why not use code to express emotions? Why not unify the language of code, the language of music, and the English language into one work of art? Why not make the artistic process itself the essence of a performance? This "why not" gives me freedom as an artist and fuels my never-ending passion to keep learning, creating, and sharing this creative space.

https://www.earthtoabigail.com

Photo by Jorge Ramirez

Photo by Steve Welburn

MicoRex
Mexico

Jorge: When in architecture school, I started live coding, influenced by diagrammatic thinking and macro-micro relationships. You build up a system in a musical context, and whatever you throw in affects the perception of the whole piece; also the system.

Tito: Live coding is an opportunity to get involved expressively and theoretically with the people who attend the live show and with other live coders. I love live coding because it gives me the thrill of present tense, feeling, language, and thought.

T: Before MicoRex I started promoting live coding as an academic practice in the Mexican electronic art circuit through the Centro Multimedia.

J: MicoRex ended up being very influential for lots of acts that came afterward since live coding and bands were not existent before that time. It helped to free experimental coding from electroacoustics and present it as a joyful, fun, and edgier practice.

J: When we started becoming international, we found acts that were trying to bring live coding into the club realm, the first generation of algoravers—like Alo Allik, Glitch-Lich, slub, Shelly Knotts, Benoit and the Mandelbrots, Norah Lorway, Kraftwife, Frederik Olofsson, Renick Bell, Luuma—like-minded acts, but everyone had very different styles and systems.

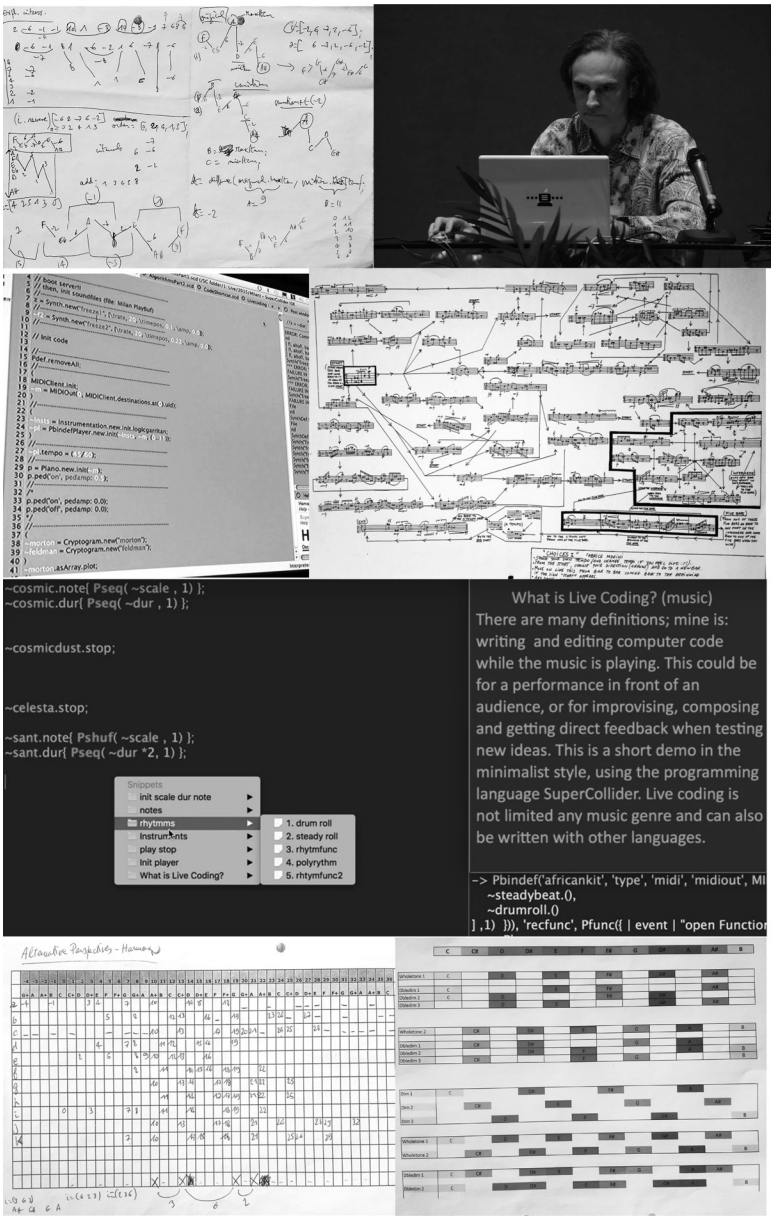
T: As Mexican live coders, we all shared a common minimal style since we came out of the same school of practice. Eventually, each live coder found their own programming style, and MicoRex would distinguish itself from the others in the performance sense: the inclusion of voice, DIY controllers, and more or less closed traditional musical structures.

J: Our system is designed by thinking about the music we want to create. Music first, technology after. It relates to other systems in goals but not in design. Elevator pitch: it's a live coded, networked, audiovisual act with voice and physical interfaces triggering OSC messages through a GUI system.

T: We do synthesis, not sampling. And, though we come from a "from scratch" live code scene, we also use ready-made algorithms. We don't use a messaging network system like Republic or MandelHub. We just send the raw data and trust our musical and supernatural monkey abilities. More like when playing in a band because, oh!, MicoRex is a band.

J: It's mind-blowing to witness a kind of normalization in a nightclub setting. I experienced violent reactions to live coding for dancing in Mexico! Now, the global network of live coding parties is becoming more established. There are more systems and tools, and so acts are becoming more easily expressive.

T: Yet live coding has limits as an expressive tool. You can't prepare all the musical aspects you want to present. Live coding has not this aim at all. It's an opportunity to play in a way in which improvisation and knowledge are more appreciated than a perfect final presentation. It is a phenomenon of our time and capabilities.



Live coding and using charts for computer-aided composition.
Image credit: Fabrice Mogini

Fabrice Mogini

There are many definitions of live coding; mine is: writing and editing computer code while the music is playing. This could be for a performance in front of an audience or, in my case, for improvising, composing, and getting direct feedback while testing new ideas.

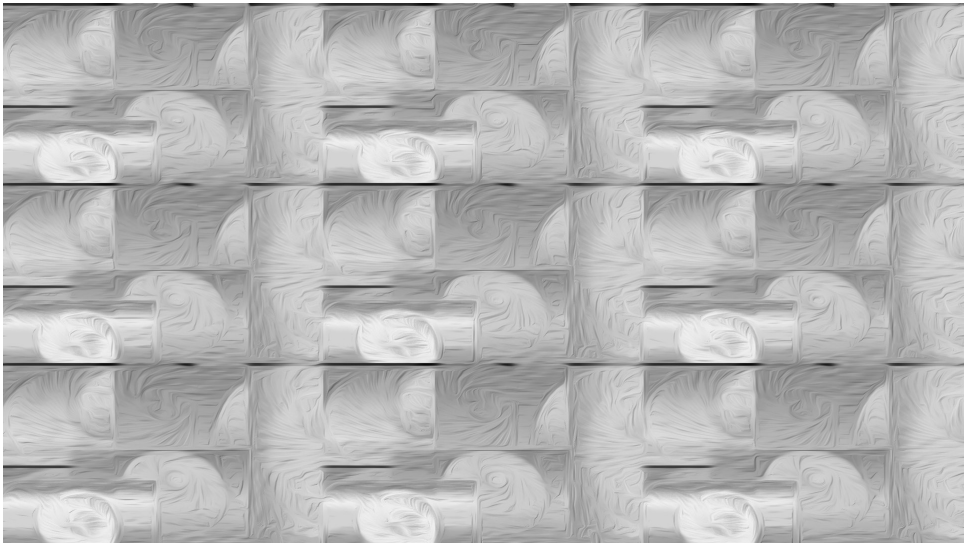
Before I started using an audio programming language, my compositions required a lot of calculations using charts, grids, a deck of cards, and so on. I could not believe my luck when around 1998 I gained access to SuperCollider 2.1 while studying for the Sonic Arts degree in London. My dream of fast and precise calculations was finally becoming true! The main problem was that once I had programmed something new—for instance, a new part in the middle of a composition—I had to wait until it actually played to hear and assess the quality of the edit. Because the music was generative, it was not easy to fast-forward, as can be done with a sequencer. Another problem was that while the expanded computing capabilities allowed ever-more complex calculations, there were now too many parameters to be serialized. The music was becoming too much of a mental abstraction rather than based in real-time perception. This is when I embarked on a quest to change and eventually write code in real time so I could hear what I was designing straightaway. Rather than just switch algorithms or edit certain parameters, I was trying to write some of the code in real time. Although I incorporated some of these techniques in live performances, my main incentive for live coding was to expand compositional capabilities while preserving real-time control and feedback. I also coded jazz improvising algorithms ("Memory" in *Morpheus* CD-ROM) and realized that live coding is a necessity for truly live improvisation.

I started to use the expression *live coding* around 2002; I remember by then that Alex McLean had already used some form of live coding in his performances with slub. In September 2003, I advertised my live coding set with SuperCollider at the 291 Gallery, Hackney, London. While the set was based on controlling, editing algorithms, and sending data in real time to Director/Lingo for visuals, there was nevertheless a part for live coding. By then I had also created the London SuperCollider course (2002–2004) at Rising Tide, London, which in 2003 included a lesson with a handout titled "Live Coding." This handout was later presented during a workshop at the IDM (Intelligent Dance Music) Summer school, London, where I had been invited as guest lecturer. At that time I often performed with composer/researcher/live coder Nick Collins; our performances, notably, included a live coding duet at the Royal College Art Bar, London, in June 2003, with real-time visuals by Fredrik Olofsson. Finally, many of our SuperCollider "tricks" and fixes were solved when researcher and performer Julian Rohrhuber improved the SuperCollider language to make live coding more accessible. Only then did I realize how many practitioners were showing an interest in that art form.

I am amazed to see how live coding has evolved in the last twenty years. There are still so many different avenues to explore, and I hope to see a future where coders won't be overwhelmed by the technological aspects but will also create beautiful music.

www.fabricemogini.com

Livecoded Splatter in Livecodelab.
Image credit: Kofi Oduro (Illestpreacha)



Altered live coded piece.
Image credit: Kofi Oduro (Illestpreacha)

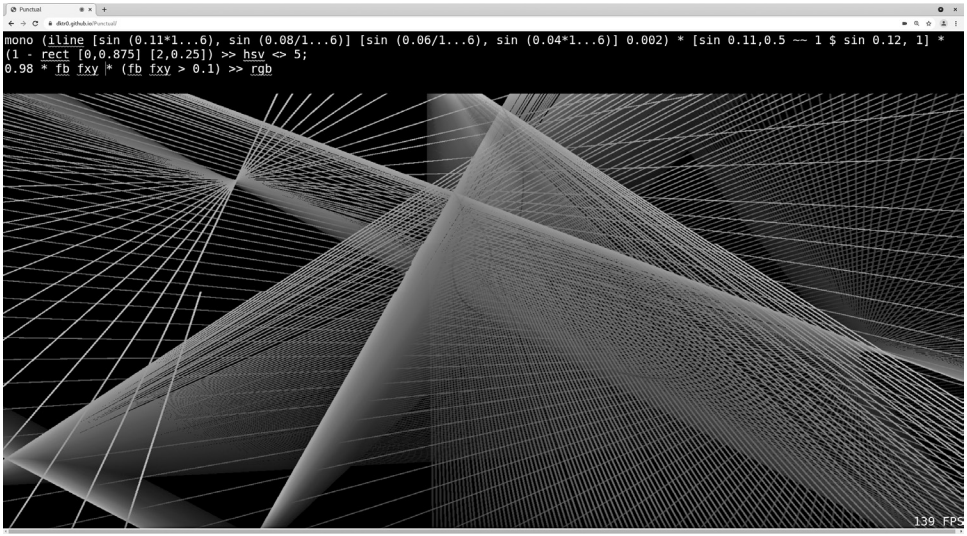
Kofi Oduro (Illestpreacha)
Montreal, Canada

Live coding to me is giving a digital extension /
Of myself, where it has a new form
A form of digital representation / A form that
breathes through the environment
Where it can mimic a therapist / A scientist or
simply just another experiment
Of playful interaction / Or meaningful actions
that leads
To both addition and retraction / Retracting
the mind to a place
Where the mind breathes / Through the bits
Giving clues as the feedback / Gives what the
program believes
From the keystrokes that are being hit / The
mathematical equations
That are feeling split / Where the emergence
of creativity
And randomness meets up quick / As the
mind goes through its own stack
Merging the elements / From numeric to digital
Analog to pivotal / Seeing how the lines of
code/ When given the chance to roam
Flows through the ears / Through the eyes
they visit
Where the variables / Are given in doses that
are edible
As the goal is to be laid back in swarms / As
not to give the senses a storm
But rather allow for curiosity / To emerge like
rain does to a worm

Live coding to me is just like this poem /
Pieces adding together
To paint a picture for worse or better /
Jamming with myself or others
As we are on a journey below and swim
Swimming with imagination
Improvisation with no intimidation
Where the culture allows for conversations
With no limitations
To the ideations & creations
That help the ideas be their own levitation
As they signal from the head
To the fingertips
And if vocalize, through the passage of the lips
As the vibration in the air
Mixes with the intent
Blends with the indents
That can be seen on here
When a spark is needed
It provides the flare
When a thought disappears
It reappears with a vision that is clear
As needed

Poetry is Code
Code is Poetry
Code occurs outside the computer screen
As it comes thru the journey, a human seen
And put together is the nature of this live
coding scene
For these are my thoughts, when asked to reflect
Of live coding, which I came to respect

https://portfolio.illestpreacha.com/links
https://colorscape.illestpreacha.com/
https://instagram.com/Illestpreacha



Punctual live coding language.
Image credit: David Ogborn

David Ogborn
McMaster University, Canada

To define something is to stake a claim to its future, to make a claim about what it should be or become. This makes me hesitate to define live coding. Forced to choose from a number of "operating definitions," the one that gives me the least anxiety is to think of live coding as the "theater of code." This is not only a theatrical space constructed from code but rather a theatrical space that thematizes code (and computing and software), asks questions of these things, exposes the skeletons in the closet, and maybe even allows for catharsis, reconciliation, and new visions of collective possibilities. This can sometimes be iconoclastic, but it need not be. With repetition, even simple interventions in the machine gradually become larger shifts. To constitute live coding as theater is to resist constituting it instrumentally (i.e., as simply about producing things more efficiently or in novel ways) and to insist on an irreducible role for representation and interpretation.

My path around live coding has been influenced most immediately by my role as the founding member of the Cybernetic Orchestra, a laptop orchestra at McMaster University that has been meeting and performing continuously since early 2010. That, as well as my teaching in the Department of Communication Studies and Media Arts, puts me into almost daily contact with situations in which different people are encountering and responding to live coding in different ways. At the same time, live coding is a site where personal interests of longer standing intersect: live electronics, improvisation, philosophy, education, free software, generative music, politics, and maybe even science fiction. My live coding activities take different forms: performing by myself or in a duo with either an autonomous agent I am cultivating (Daem0n) or with tabla player Shawn Mativetsky (as very long cat); weekly Cybernetic Orchestra rehearsals; developing and maintaining multiple software platforms; writing about these things. . . . Perhaps my most obvious contributions will be as a popularizer, as I have introduced many thousands of university students in large survey courses to live coding over the years. I hope to contribute to establishing interpretation and critique as elements of live coding culture.

The software systems I have been developing in recent years are strongly oriented toward collaborative, geographically distributed live coding. Estuary is a sprawling web-based platform for live coding, developed with the support of two grants from Canada's Social Sciences and Humanities Research Council (which also strongly supported early research around the Cybernetic Orchestra and the second ICLC in 2016), and "hosting" a growing number of independently developed live coding software projects (thanks to the magic of open-source licensing), including my pet project, the audiovisual language Punctual. The idea of supporting and mixing up multiple languages has been a fixation in my technological work (earlier, for example, I worked on a "language-neutral" synchronization system for live coding ensembles). This is motivated by the idea that monolinguistic cultures and ethnocentrisms are mutually reinforcing. Live coding, performing a theater of linguistic plurality and complexity, may create a space for less anxious, more welcoming futures.

http://www.dktr0.net

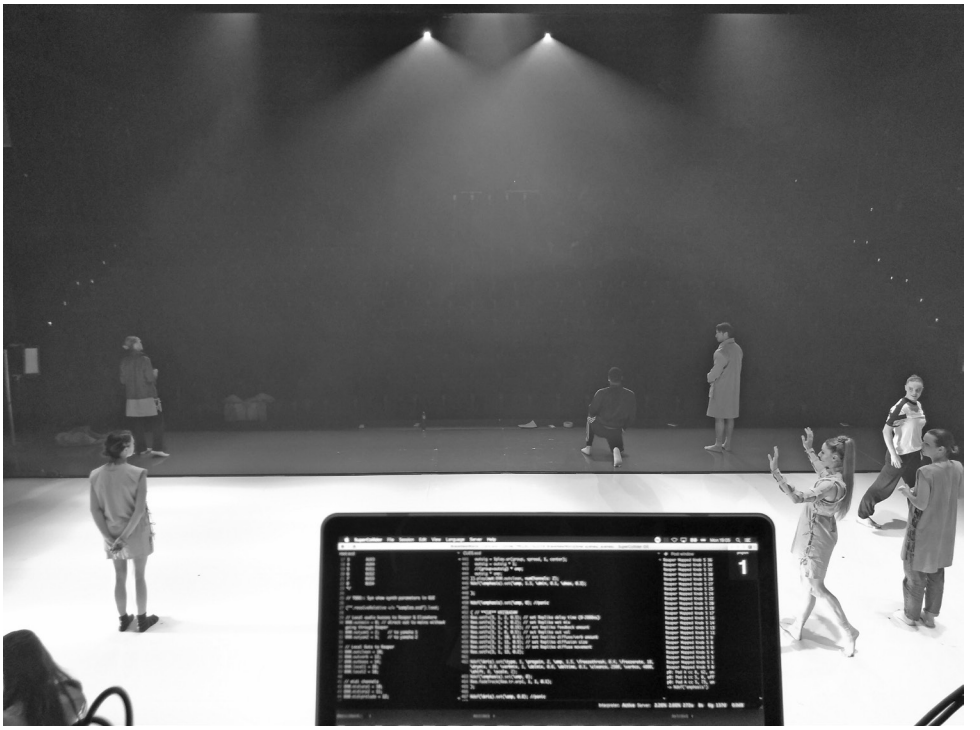


Photo by Jojojo star

Photo by Jojojo star

Jonathan Reus
University of Sussex, The Netherlands/UK

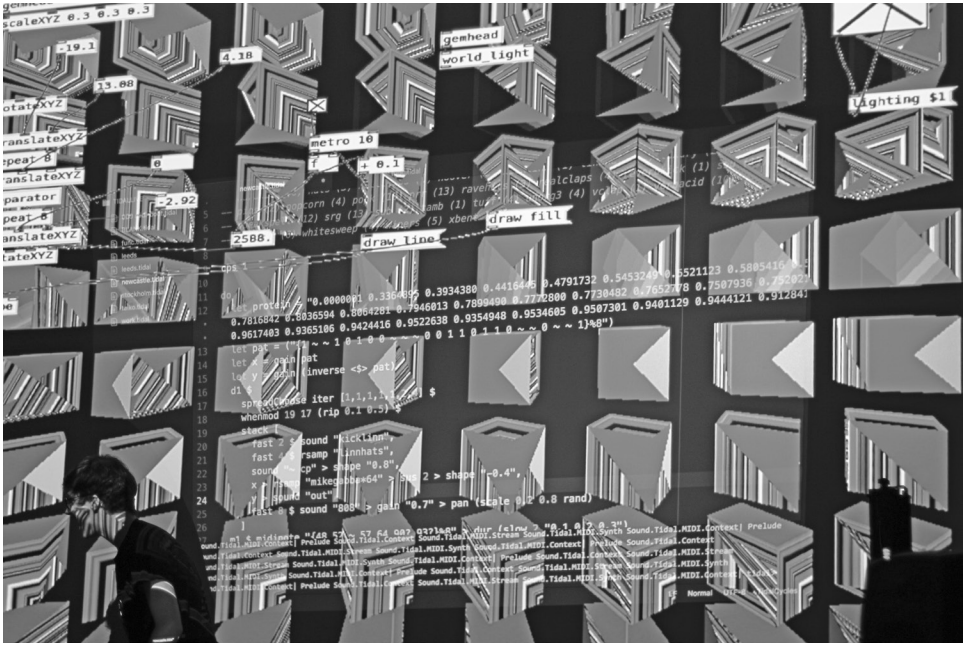
My first encounter with a live coding performance was at the Studio for Electro-Instrumental Music around 2012 or so. I remember finding it intriguing but also kind of impenetrable. For me live coding tends to be a way of working and thinking creatively rather than exclusively a performance practice.

I started using SuperCollider years ago when Marije Baalman and I started a music/programming meetup group in Amsterdam. Marije was at the time a major contributor to the SuperCollider code base, and she made a pretty compelling case for using it. I found it to be an expressive and immediate way to sculpt ideas in time, but I've never thought of myself as live coding. It's more that the act of making anything using an interpreted programming language involves live coding in some sense, be it an installation, a piece of music, or a light-control GUI for a theater production. The creative process is more like sculpting than engineering. You build up a little bit here. You remove a bit there. You work in small gestures, not entire programs.

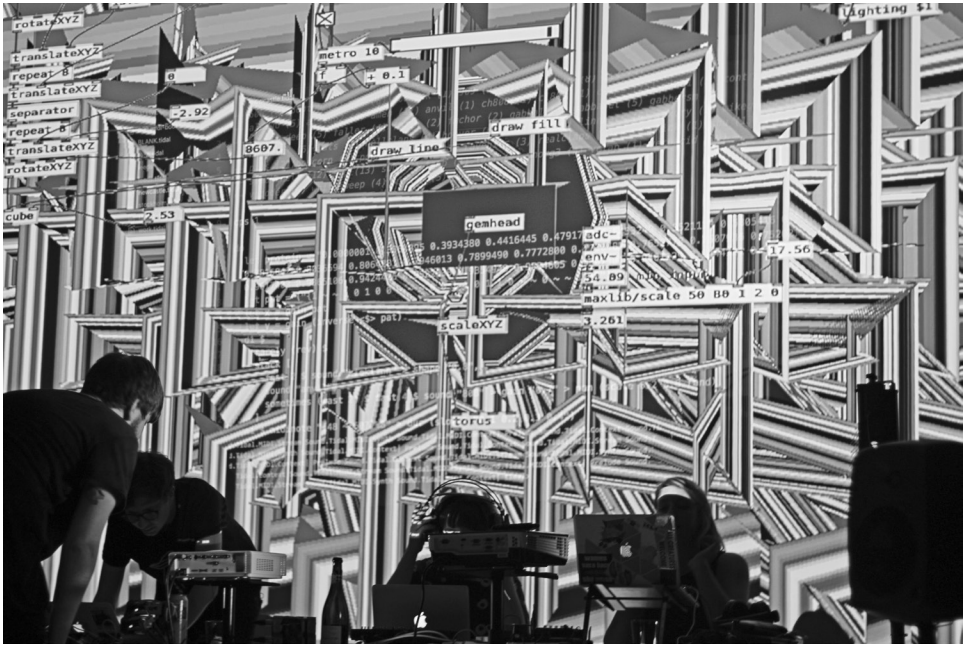
I have performances that involve pulling apart computers while they are running different software where all the sound material is generated from a combination of the metal, plastic, and electricity of the computers. Live coding makes an appearance in these performances as a dramaturgical and narrative element. I might run epistolary commands in the terminal or write missives in a text editor, and these add to the storytelling of the performance. Other work, like *Anatomies of Intelligence* (my collaboration with Joana Chicau), goes so far as to use live coding as a kind of epistemic philosophy to engage with larger themes, such as the spectacle and theatrics of scientific knowledge or bias in the data corpuses of AI systems. Anatomies of Intelligence uses a bespoke system that exists within a web browser. In the latest version of this performance, we created a "Virtual Theater," a website that Joana and I remotely access and relay JavaScript commands to. Everyone who is visiting the site gets our JavaScript commands relayed to them and executed in their browser, directly manipulating the web page, including graphics and sound. We compose a very specific narrative for each performance, with room to improvise, and perform together using a shared text editor.

I'm hoping that live coding stays idiosyncratic, and that the community continues to grow. And, especially, that artists and toolmakers think outside of established audiovisual performance norms and communities. Reach out to unusual audiences, remix, and collaborate. Recently, I organized a Zoom aerobics workout class, the ALGO-RHYTHM DANCE WORKOUT, with a dance instructor teaching a routine to live coded visuals and music. It was beautifully absurd and absurdly fun.

jonathanreus.com



Chemical Algorave
Photo by Antonio Roberts



Chemical Algorave
Photo by Antonio Roberts

Antonio Roberts
UK

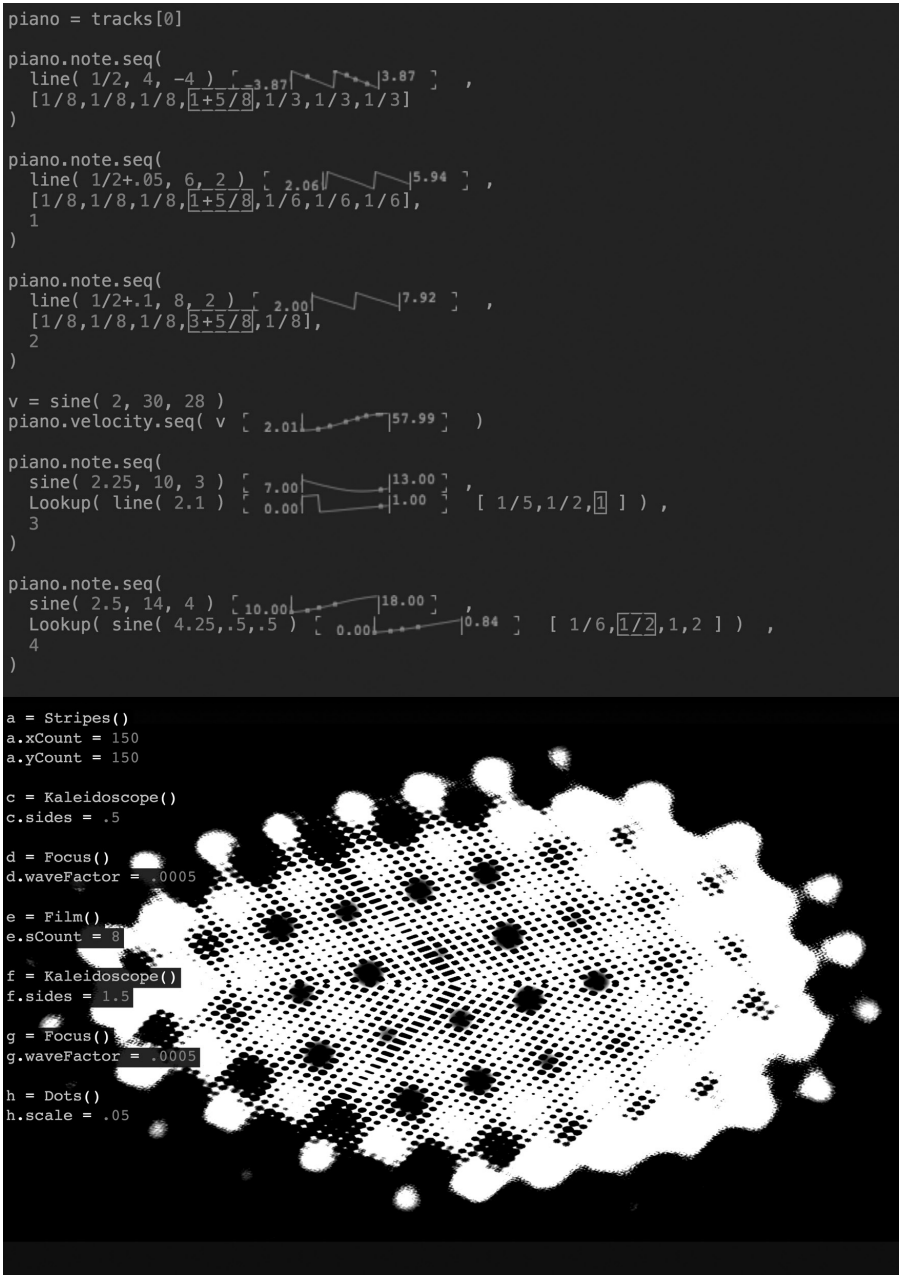
During my time with BiLE (Birmingham Laptop Ensemble) I had already started making live visuals for electronic music, although this was more a case of manipulating existing software and code. Knowing people such as Shelly Knotts, who had already performed at algoraves, gave me an entry point to start live coding in early 2014. At that time there were relatively few people live coding visuals, and so I found myself in a unique position.

My live coded visuals start off with a simple geometric shape, usually a cube. I might then start to manipulate that one cube by rotating or scaling it. Then I will multiply the amount of cubes and have them spin in different ways, changing colors or changing size depending on other factors such as the amplitude of the music. I try to code and present this in a way that is clear enough for the audience to see how I arrived at the result on-screen, but inevitably, it ends up looking messy!

Building your code slowly over time is not a bad thing. In the thirty minutes or so you typically have for a set, you can build up your visuals slowly or scrap it all and start again. Don't exhaust your code and yourself by building up all of your code within the first five minutes.

Both VJing and live coding music have a strong visual element, and I feel they both seek to demystify how computer art is made by showing the process. I have definitely seen a tighter integration of music and visuals as live coding practice has matured. For example, artists now have visuals that react to the music in more ways than just amplitude. Some programmers are even building visuals into what was initially only audio software.

http://www.hellocatfood.com/



Top image: Annotations in Gibberwocky show waveforms that are periodically sampled to generate musical patterns. Bottom image: Multiple post-processing shaders stacked in Gibber to create an abstract form.
Image credits: Charlie Roberts

Charlie Roberts
Worcester Polytechnic Institute, US

Gibber is a browser-based environment for audiovisual live coding. It primarily uses JavaScript as the end-user language while offering affordances for both music and graphics programming. A dedicated server supports user publication of sketches, real-time chat, and a variety of other collaborative features. A trio of derivative live coding systems coauthored with Graham Wakefield—collectively named Gibberwocky—borrow many aspects of Gibber's interface to target external applications, including Max/MSP/Jitter, Ableton Live, and generic MIDI communication.

When I initially began work on Gibber, I was already working extensively with browsers and JavaScript, creating end-user frameworks for interface design. I wanted to explore the potential of JavaScript as a live coding language and the potential of the browser as a vehicle for the collaboration and dissemination of creative work. By chance, at the time these interests were coming together—in 2012—the Web Audio API was released for browsers, enabling real-time synthesis in the browser using JavaScript alone. Gibber was one of the earliest systems created for live coding performance in the browser; now there are a variety of excellent options.

Running in the browser makes Gibber an easy match for introducing people to live coding; no extra software is required. This has helped enable workshops all over the world that use Gibber to teach the basics of live coding and computational media. A variety of summer camps, after-school programs, and university courses have also used Gibber heavily.

One significant area of research I have been exploring with Gibber/Gibberwocky is the use of dynamic annotations and visualizations in source code that documents the state of running algorithms; many of these are shown in the top image on the left. Heavily inspired by Thor Magnusson's ixi lang, the feedback provided by annotations/visualizations is typically what I receive the most positive comments about after performances; perhaps with more practice someday audiences will respond similarly to the music I create.

Gibber seems to have found a niche as a live coding system for beginners. Its reliance on the browser makes it easily accessible but also stops advanced programmers from using their preferred code editor for performance—and the editing interface is, of course, a critical component of live coding. I hope to eventually attract more advanced programmers via Gibberwocky, which provides unique integrations with artistic production software not found in other live coding systems, and also by porting Gibber and Gibberwocky to run inside other editors. Gibber and Gibberwocky have drawn inspiration from a number of other live coding systems, especially ixi lang, Tidal, and Extempore.

The continued development of Gibber/Gibberwocky is highly dependent on my performance practice. Typically, I have an idea I'd like to explore in a performance—perhaps musical, perhaps an interaction technique—and this spurs subsequent development work. As an academic, the research potential of new features is also an important consideration when deciding which elements to focus development on.

http://gibber.cc

Screenshot from live coding performance INVOCACIONES.
Image credit: Jessica A. Rodriguez

Jessica A. Rodriguez
Mexico/Canada

Invocaciones ("Summonings") is a series of performances that arise from the need to explore my artistic practice through the voice of female poets, mixing live coding practices (or code on-the-fly) with electronic literature (or expanded literature practices through digital environments) and exploring the poetic possibilities of code and what speech it activates.

For the performance *Invocaciones*, I remixed the poem "País de la Ausencia" by Chilean writer Gabriela Mistral, which explores images about identity, place, and territory. Through the author's voice, I made a reinterpretation of the poem, expanding her words through different sound layers that move through a stereophonic space. I use TidalCycles to create speech patterns (transforming voice samples) through cacophony, juxtaposition, delay, echo, and oversaturation to deconstruct and reconstruct the poem over and over again throughout the performance.

Additionally, I use prerecorded cello samples by Mexican-Brazilian cellist Iracema de Andrade. The visual part contains a video (running in the background) of the Paricutín volcano in the state of Michoacán, Mexico.

https://andamio.in/production
https://vimeo.com/jessicaarianne

Screenshot of live coding with TidalCycles and Hydra.
Image credit: Iris Saladino

Iris Saladino
CLiC (Colectivo de Live Coders)/TOPLAP Argentina

I am a sound-oriented multimedia artist. I live in a world in which in order to make a living out of art you must conform to the market, the "culture industry."

Live coding is a practice born in academic environments but now exceeds them, flowing into groups of people with different backgrounds who voluntarily study and share knowledge, mainly in digital spaces. Invested with ideas from free software culture, live coding creates new flows of information and power, challenging notions such as verticality, hierarchy, professor, pupil, author, creator, artist, technologist, talent, idol, and art. Communities operate by decentralizing and horizontalizing information, processes, events, and decisions with respect for others as the core of all interactions. We aim for collaboration and we mistrust competition.

I am a member of CLiC, based in Buenos Aires, Argentina. It started as a small group and now is a large one. Cyberfeminist oriented, this community is where I discuss the most interesting topics, such as technology, ethics, politics, aesthetics, philosophy, and art theory among others, which are all very important to me as a creative person. We do not always agree, but it is nice to deal with differences when the environment guarantees tolerance and cordiality. We share technical and scientific information, and we help each other to solve code or tech problems. My consciousness of the context I live in increases thanks to this group, and with it my actions in the world, my artwork included.

I live code my music using SuperCollider and TidalCycles and my visuals using Hydra. When coding sound, I usually feel like creating floating, circular moving structures: sounds emerge and disappear and change position, duration, and timbre; the whole combination creates convergence and divergence on different levels. I become entranced creating those structures, making them turn, imagining processes, and typing them, exploring audio samples and digital synths to their limits. I play and there is a pure joy in which the rules are not for winning (success over others or other's approval) but for sustaining the development of the activity across time, just like a child's free imagination.

I often jam with live coders from around the world using flok, a collaborative peer-to-peer online editor. Even when not sharing a physical space, sound keeps allowing us nonverbal communication by interacting with coded processes. We hear, complement, and understand each other, we answer the sound proposals of our colleagues, we learn from each other, and we have a lot of fun. We can embrace error as imperfect beings, recognizing and enjoying our diversity.

The whole thing I describe composes a poetic, new model for art and social interaction in digital environments. It fills me with hope for the future.

https://iriss.netlify.app/

Photo by Kamil Kurylonek, 2014

Kate Sicchio
Virginia Commonwealth University, US/UK

My main live coding practice focuses on live coding performance scores for dancers performing choreography for the stage. Dancers improvise movement based upon instructions that are projected into the performance space. I have explored this in many ways, including using pseudo code to create rules for dancers, live coding haptic feedback in costumes, and using machine-learning algorithms and an image database to create a visual score. This work is typically shown in black-box theater spaces as a live, improvised performance.

I started live coding after exploring the concept of computer hacking as a way to repurpose and to extend this repurposing beyond technology. If I could hack my kinect to work with a laptop, could I hack a piece of choreography to change the intended outcome? *Hacking choreography* became an umbrella term for many works I made in which a choreographic score would be changed live in a performance setting.

I started this work before discovering the live coding community. Once I saw the TOPLAP manifesto, I realized this reflected my choreographic work and underlying ideas. The use of the projection to see the code and the changes, the live interpretations of the instructions, and the performers thinking and making decisions as part of the performance were all things I also wanted to highlight in the work. From here I went on to create two different programming languages for live coding choreography, Terpsicode and Studio//Stage.

I have collaborated with other live coders in different ways. Rodrigo Velasco has composed sound for my work, and Nick Rothwell collaborated with me on creating a system for live coding clojure to create animated text. I worked with Thomas Murphy on a live coding environment for images to create a visual choreographic score. But a much more in-depth collaboration has been my piece with Alex McLean, *Sound Choreographer <> Body Code*. This work creates a feedback loop between McLean's live coding of sound and a generative choreographic score I am performing through using sound analysis and motion tracking. These technologies connect our two forms of improvisation and affect our decisions within that performance.

Currently, I also live code music as one-third of a trio, named Codie, with Sarah Groff Hennigh-Palermo and Melody Loveless. We wanted to become more involved in the algorave scene and saw the formation of Codie as a way to participate. What is interesting about Codie is that our audiences always dance to our sets. So, despite not explicitly coding instructions for movement, we have managed to create performances where people are dancing to code, as found in my choreographic works.

http://sicchio.com/

Photo by Clément Merle

th4
France

For me, live coding is not just creative coding, as it involves some kind of live component, which could range from completely improvising a piece of artistic code onstage to tweaking an already prewritten algorithm. I came to live coding in a fairly traditional way: I come from a computer science background, and I wanted to make music. DAWs were a bit obscure to me, and I felt that my prior knowledge of programming would give me some kind of head start to make up for my relative inexperience in music. I almost exclusively work with TidalCycles, and I feel like that scene is more oriented toward the rhythmic aspect of the practice, while mine is much more on the harmonic side.

In the future I think we're bound to see AI play some kind of role in music in general, particularly as a tool to generate unexpected ideas, and it would greatly surprise me if this didn't make its way to live coding in some way or another, even though I don't imagine it completely taking over and leaving no space for more "handmade" algorithmic creation.

From my experience performing in front of non-code-literate audiences, the visual aspect of a live coding performance can exert some fascination on the general public from the esoteric aspect of lines of code that you don't understand giving birth to a piece of music or visual art. This can also, on the other hand, be a good pedagogical starting point to break down the code into something understandable.

https://th4music.net

Photo by Cihad Caner, reworked by Felipe Ignacio Noriega

Anne Veinberg and Felipe Ignacio Noriega
CodeKlavier and Off<>zz

Overture: Anne Veinberg (Aus, NL) is a pianist, and Felipe Ignacio Noriega (MX, NL) is a composer and programmer. Since 2013 they have formed a music and research duo, Off<>zz, with the mission of bringing live coding into the classical music sphere. In 2017 their main focus shifted from primarily performing as a duo to creating a fused practice in which one makes algorithms by playing the piano. This project is called the CodeKlavier. The CodeKlavier employs the piano as an instrument for live coding and is influenced by musical thought to drive programming language design.

Thema Adagio: The CodeKlavier spawns from treating musical gestures on the piano as the syntactic sugar that can generate code constructs such as variables, snippets, conditionals, and function definitions. The implementation of the CodeKlavier is strongly based on functional programming concepts and music analysis. It comprises three main areas: (1) The parsing of piano playing into programming expressions, otherwise known as the *piano parser*; (2) The creation of algorithmic structures from the parsed building blocks; (3) The code outputs, which can be developed on any live coding platform. The latter allows us to collaborate with different creative coders to "piano code" on a wide variety of artistic planes. By 2020 these collaborations included code output extensions developed by Patrick Borgeat, Timo Hoogland, Joana Chicau, and Sebastian Pappalardo.

Allegro variazioni: Although mostly associated as a duo and for their work on the CodeKlavier, their individual projects are also intimately involved with live coding. Felipe has been exploring live coding and humor through the hip-hop band Panda Zooicide, in which live coded beats and a rapper explore the relationships of freestyle rap and the improvisatory elements of the live coding practice. Next to these projects, he also employs live coding extensively in all his output as composer: mostly music-theater pieces for children with the Norwegian-Dutch theater collective Krims-Krams and other collaborations within the classical music sphere. Anne is a professional pianist. Her live coding practice is always connected with piano playing whether it be piano coding with the CodeKlavier, collaborating with other live coders, or attempting to multitask and code via laptop while playing the piano.

Coda: When thinking about live coding, we expect different things. Anne often looks for a fulfilling musical or artistic experience, while Felipe seeks out humor and transparency in the code. We both believe in the powerful connection that the audience members make with the performers when they are able to follow the development, thoughts, and decision-making of the artists and how the performance environment influences all of these. Live coding is a medium that can reflect and highlight this relationship, and that is one of the main reasons why we are seduced by this practice. We also believe that musical intuition and a deeper understanding of music aesthetics and theory can unlock new approaches to technological development.

https://codeklavier.space
https://keyboardsunite.com/offzz
https://pandazooicide.com

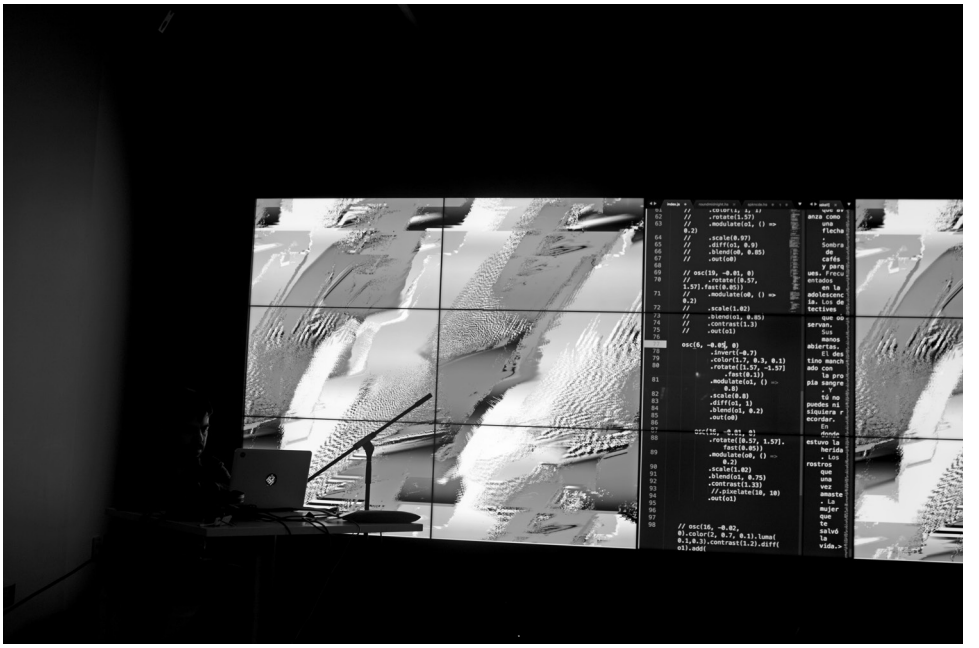


Photo by Ali Barilaro

Rodrigo Velasco (yecto)
Mexico/Canada

First, I want to thank all of those who are part of the live coding community, which is an example of openness and diversity. Live coding helped me escape. I consider it a back door or an escape tunnel to reimagine and relearn how to feel and rethink language and programming. In 2011 an interest in escaping from graphic design, making music, and exploring or reimagining both through the use of open-source software fortuitously led me to meet an incredible community in Mexico City. It met in the open-source software workshops and the monthly live coding sessions that took place in the Galería Manuel Felguérez of the Centro Multimedia and were organized by El Taller de Audio del Centro Multimedia. I remain grateful to everyone in the community, especially to those who patiently shared ideas and knowledge with me across and beyond live coding.

An essential aspect that live coding has gifted to my artistic practice is the idea of "algorithms as thoughts."[10] I am interested in poetics and algorithmic poetics, a confluence of language; algorithms as thoughts in movement; and the creation of a space-time that activates "a transversal movement that bonds content and expression as assemblage."[11]

I am currently studying for a master of Design and Computation Arts at Concordia University and developing a research-creation project, a living repository of Nahuatl memory, with the name *Algorithmic poetics*, which consists of an ongoing process of reimagining Nahuatl poetry.[12] Embracing Nahuatl principles expands our perspective in the study of algorithmic poetics, exploring forms of coding that transform how we experience the web.

These interests also converge, although in a different way, in the project I have developed under the alias yecto, which consists of composition and improvisation with sequences of percussive patterns and chords; a state of calm that allows one to experience peace of mind. yecto is at the crossroads of ambient, jazz, and hip-hop, but it is also live coding; the sounds are coming from MIDI signals interacting between TidalCycles and/or the esoteric programming language ORCA, with hardware synthesizers and often chopping samples or playing electric guitar. Regarding the visual dimension, yecto is actively exploring the creation of generative design and live coded visuals, mainly through Processing, p5.js, and Fluxus, and a recent undertaking has been in the study of Hydra and video feedback.

Finally, I want to thank the live coding community around the world, especially my dear friends Ernesto Romero, Alex McLean, Karen del Valle, and Olivia Jack.

https://soundcloud.com/yecto

Photo by Antonio Roberts

Elizabeth Wilson
Queen Mary University of London, UK

I see live coding as a method of notating and enacting creative ideas in a way that is fundamentally human—through language. For me, this most commonly takes the form of creating music, converting musical ideas from the graphemic to the acoustic. Alternatively, I like the definition of live coding as "writing a 'score' for the computer to perform,"[13] as it shifts the perspective from solely human control to more of a symbiosis. The live coding language I predominantly use is TidalCycles. Being based in Haskell, which is a purely functional language, it is easy to build around and augment code by constructing your own functions. Tidal is well designed in its ability to express complicated ideas with compact language, often closely resembling sentences. Because Tidal is capable of a broad scope of expression, I want to extend the creative opportunities afforded to a live coder by building autonomous agents to perform with.

I am really drawn to the idea of being able to share creative responsibilities with the computer. I've always wanted to avoid the constraints of gestural control that come with most musical interfaces. I found that automating processes previously requiring manual skills leaves more mental capacity for traversing unexplored areas of creative spaces and uncovering new territories of ideas. My own research involves incorporating *affective response* into musical generation, mainly in TidalCycles. This is an important consideration for any future autonomous systems in live coding. Approaching the music generation task from a purely computational standpoint detaches it from its essence of inherent emotional expression. This seems obvious for automatic text generation, which considers the narrative and its intended message rather than solely syntactic information. It follows that the same should apply to music and that improvising with machines should be an exchange of meaning.

I'm also inspired a lot by the work of Renick Bell with the Conductive system. I like the way that his "players" seem somewhat alive; that they can get bored of a pattern playing and take over and change things. I often think about this idea of sharing creative responsibility, particularly when performing live. Sometimes things might become overly repetitive halfway through constructing another pattern, so having a machine partner to enact some of the responsibility could resolve these issues. I think we're only just beginning to scratch the surface of how to utilize these kinds of collaborations, especially if we can view destruction as a form of creativity too.

Many experiments in AI have been recreating the works of composers who have been dead for a while, and not enough are creating new music to dance to, arguably one of music's most important functions. However, there's a significant need for more stringent consideration of the ethical implications of where and how we use AI in live coding, which, surprisingly, can still be overlooked. Algorithms are often viewed through the lens of how they are used by large corporations. The word *algorithmic* in itself is not dangerous but can often be seen that way because algorithms can be misused to enforce discrimination or prejudice. My hope is that algorithmic music can help to change the public's opinions on algorithms, allowing them to see how they can be used for things as transformative as art and music.

https://lwlsn.github.io/digitalselves-web/

Photo by Helena Coll

Anna Xambó
De Montfort University, UK

I see live coding as a meeting point between code and music in live performance. With some classical training, and after several years of being in Barcelona-based bands as a bass guitar player, singer, and composer, I started to make experimental electronic music in the mid-2000s. In the exploration of new sounds and the boundaries of the musical language, it has been a natural turn to approach experimental electronic music from a live coding perspective. Live coding brings a DIY approach to building and sharing self-built tools/environments, along with projecting your screen, as stated in the seminal TOPLAP manifesto, while providing an algorithmic approach to performing in which each performance can be different even when based on the same code. Furthermore, the community is unique and formidable.

My contribution to the practice is as a practitioner, academic, educator, and curator. As a practitioner I've been moving from audio synthesis to sample-based music, where I try to explore its narrative using my own as well as crowdsourced sounds. My academic research encompasses inspecting different possibilities of collaborative music live coding (and its related political negotiations) and multichannel experiences with live coding. As part of my teaching, live coding has been useful for demonstration, as well as to promote teamwork and participatory activities in class, including online during this COVID-19 pandemic. As a curator I have been co-organizing live coding concerts in Barcelona (Spain) and Atlanta (US), contributing to the local and international experimental electronic music scene.

My work has been inspired by the first generation of SuperCollider live coders and the SuperCollider community, especially the Barcelona orbit of the Music Technology Group at Universitat Pompeu Fabra, as well as the informal SuperCollider workshops and meetups organized by Gerard Roma and l'ull cec. My practice has been strongly influenced by Gerard Roma's work, starting with a collaboration in the duo Pulso performing with a custom environment he wrote, inspired by Thor Magnusson's ixi lang, and using two code editors in sync.

The democratic and self-organized approach to collaborative live coding proposed by Alberto de Campo, Julian Rohrhuber, and others with the Republic system is also of great inspiration. I have explored group improvisation with Nela Brown's Female Laptop Orchestra (FLO), along with others, where my live coding was combined with a variety of other digital and acoustic instruments. Knowing of Shelly Knotts et al.'s work with OFFAL was stimulating, as well.

I have been working with the Music Information Retrieval in Live Coding (MIRLC) system since 2016, and my recent work is on a follow-up system named MIRLCAuto (MIRLCa), a virtual agent for music information retrieval in live coding. The latter explores the theme of AI and autonomous agents in live coding. Similar live coding systems include Nick Collins's "algoravethmic" remix system and its approach to live coding and machine listening, as well as Navarro's Cacharpo virtual coperformer. The web-based system Sema developed by Bernardo, Kiefer, and Magnusson, which is an ecosystem of live coding languages and machine learning, is also an enlightening project.

I envision the future of live coding as a community that will keep growing in diversity and will keep advancing alongside other related fields. Feminist and decolonizing approaches to

live coding can also open the floor to new, interesting voices and ideas. The combination of code and music seems to either attract or scare the general public. I would say that the audience's degree of computer and programming literacy often affects the understanding of a performance.

annaxambo.me

The MIT Press would like to thank the anonymous peer reviewers who provided comments on drafts of this book. The generous work of academic experts is essential for establishing the authority and quality of our publications. We acknowledge with gratitude the contributions of these otherwise uncredited readers.

This book was set in Stone Serif and Stone Sans by Westchester Publishing Services.

Library of Congress Cataloging-in-Publication Data

Names: Blackwell, Alan F., author. | Cocker, Emma, author. | Cox, Geoff, author. | McLean, Alex, 1975– author. | Magnusson, Thor, author.
Title: Live coding : a user's manual / Alan F. Blackwell, Emma Cocker, Geoff Cox, Alex McLean, and Thor Magnusson.
Description: Cambridge, Massachusetts : The MIT Press, [2022] | Series: Software studies | Includes bibliographical references and index.
Identifiers: LCCN 2022008717 (print) | LCCN 2022008718 (ebook) | ISBN 9780262544818 (paperback) | ISBN 9780262372626 (epub) | ISBN 9780262372633 (pdf)
Subjects: LCSH: Computer programming—Philosophy. | Agile software development. | Creation (Literary, artistic, etc.) | Algorithms—Psychological aspects.
Classification: LCC QA76.6 .B5794 2022 (print) | LCC QA76.6 (ebook) | DDC 005.1301—dc23/eng/20220527
LC record available at https://lccn.loc.gov/2022008717
LC ebook record available at https://lccn.loc.gov/2022008718