Teaching Strudel to young girls: Realizing live coding through performance practice

Citation for the original published paper (version of record):
Diapoulis, G. (2024). Teaching Strudel to young girls: Realizing live coding through performance practice. 8th International Conference on Live Coding (ICLC2024).
http://dx.doi.org/10.5281/zenodo.11350109

Teaching Strudel to young girls: Realizing live coding through performance practice

Georgios Diapoulis
Chalmers University of Technology, University of Gothenburg
georgios.diapoulis@chalmers.se

ABSTRACT

In this study, I discuss my first experience teaching young girls, 10-15 years old, how to use Strudel for music-making. The work presented here focuses on technical and logistical matters of organizing and carrying out such a teaching activity, and reports initial findings from the Creative Coding course at Chalmers University of Technology and the completion of the first batch of students. The study is based on my reflective diaries, to elicit nuanced information when organizing and teaching coding for music making workshops that aim to introduce young girls to programming. The teaching philosophy here acknowledges curiosity-based learning as a valuable pedagogical approach to student learning. The outcome of the course was celebrated with two public concerts so far. I discuss the practical aspects of organizing such a course and reflect on my diary notes. I report initial findings on how the students practiced using Strudel and discuss how young girls experienced dialogic liveness through performance practice. Several recommendations on best practices for teaching Strudel and music programming are presented.

1 Introduction

Live coding has been trending in education over the last decade (Aaron, Blackwell, and Burnard 2016; Freeman and Magerko 2016; Rubin 2013). Although different authors use the term differently (i.e., Rubin (2013) as writing programs in the classroom, whereas Freeman and Magerko (2016) as musical live coding), they all agree on the benefits of live coding in the education of programming. Freeman et al. (2019) argues that an introductory course that is engaging and expressive can empower students to persist in the field of computing. Live coding is known to empower learners to explore the musical characteristics and the musical structure (Aaron, Blackwell, and Burnard 2016). Thus, it provides a fruitful approach to cultivating computational thinking combined with artistic creativity by employing adventure and exploration (Blackwell et al. 2022).

Computational thinking is a necessary skill today, as many research disciplines incorporate specialized research using computational models (e.g., computational physics, computational chemistry, and so on). The recent developments of AI have been demanding to better understand and develop human creativity, something predominant present in artistic practices. Understanding human creativity would make it possible to simulate creative processes, commonly known as computational creativity. Modern pedagogical demands advocate for the 4C's, which center on critical thinking, creativity, communication, and collaboration (Kereluik et al. 2013).

The vast abstraction layers of a programming language offer an unconventional approach to making mental models between algorithmic structures and musical outcomes. Making connections between abstract programming structures (high-level cognitive processes) and sounds (auditory and music perception) is often not evident, but it is particularly useful for understanding creative processes in programming for the arts (McLean 2009; McLean and Wiggins 2012). However, in the case of musical live coding, the auditory and visual percepts mediate our understanding of this goal. In certain programming languages for music-making, it has become more and more accessible to make connections between musical events and code structures, as in the case of the visual feedback used in Strudel REPL, where code highlighting is linked to musical events. Moreover, the computational abstraction of live coding is further facilitated by iterative development (Freeman and Magerko 2016) and trial-and-error (Ward et al. 2004) techniques. Repetition with variation is also a common approach used to unpack the layers of abstraction within coding structures (Freeman et al. 2019).

In this article, I will discuss my first experience of teaching live coding to young girls (10-15 years old). The course was 10 weeks long, including a follow-up public live musical performance. As a teacher, my aim was to engage young girls with computer programming through music-making. The project was funded by the Department of Computer Science and

Engineering and sponsored by the gender initiative GENIE[1] of the university and the company of technology Reuseit[2]. The main scope of the project was to support and promote female empowerment of young girls, especially focusing on those with different socio-cultural backgrounds (Maric and Murali Rani 2024a). The artistic focus aimed to engage the young girls in a fun and enjoyable activity that smoothly transitions from a STEM approach to a STEAM pedagogical approach.

The project focused on developing the digital skills of young girls to cope in the ever-changing world, where new skill acquisition is required throughout the lifespan of modern humans. The main tool used during the workshop was Strudel (Roos and McLean 2023), a web-based JavaScript version of TidalCycles (McLean and Wiggins 2010). Strudel is a relatively new project, begun in March 2022, and has been evolving into a vibrant and engaging community. In this article, I will discuss the young girls' appreciation of Strudel, what worked well during teaching and what did not, and how the girls started to live code during performance practice. I also provide some insights on possible future directions of Strudel, which can extend to other live coding languages.

The current study uses as the main material my reflective diaries throughout the course. I will discuss the teaching approach used, with a particular focus on the programming side of teaching Strudel. How the girls approached algorithmic composition and understood sound production using Strudel will be central to this article, by presenting the scope of best teaching practices for the specific age group, of 10-15 years old.

2 Background

2.1 The design of the Creative Coding course

The Creative Coding (CC) course is part of a specialized track of courses at Chalmers University of Technology, Gothenburg, Sweden, aiming to target school students. The first run of the CC course was from March to June 2023. The course included a 10-week workshop schedule followed by a public performance in a technological conference venue. During the 10 weeks, were carried out 8 two hours workshops during Thursdays and two four-hour workshops on Saturdays. In the Autumn of 2023 (October-November), three more two-hour workshops were carried out for the preparation of the second public performance in the arts and science festival, AHA[3].

The CC has three pillars (Maric and Murali Rani 2024a). The first pillar is empowering the girls by offering them a chance to experience computer programming and knowledge of worth (Kereluik et al. 2013) through music-making and performing arts. The second pillar is to support the acquisition of social capital for better social integration by meeting other girls and mentors. The third pillar aims to open the doors of a well-established technical university to motivate future educational choices.

The course was designed for a maximum of 16 girls who were recruited from under-representative areas of the city, with diverse cultural heritage and sociocultural backgrounds. The course team had a senior academic as project and research leader, a research assistant, two doctoral students responsible for teaching programming, including myself, and five mentors were acting as role models for the young girls. All members of the course team are women, except me who I identify as a man. The mentors were either doctoral students or university teachers.

The course took place in the Kuggen building in the Interaction Design (IxD) studio, which is the working space of the master students of the master's program Interaction Design and Technologies, Chalmers University of Technology, Department of Computer Science and Engineering. The course organization ensured parental consent for the young girls to participate and allowed us to produce research as the outcome of the course.

The technical equipment acquired for the course was two sound mixers, two pairs of studio monitors, 10 used Chromebooks, 16 pairs of headphones, 16 soundcards, and audio cables. We provided every student with an HP Chromebook, a pair of Beyerdynamic DT 240 PRO headphones, and an M-Audio M-Track solo USB soundcard. The equipment was stored in the university building.

2.1.1 Pedagogical approach

Our main pedagogical approach was fostering curiosity and critical thinking in the young girls by teaching them how to write code for music-making (Maric and Murali Rani 2024a, 2024b). Curiosity is seen as an important quality of human cognition, and it is acknowledged to facilitate and motivate human learning (Ten et al. 2021), whilst its mechanisms are

[1] https://www.chalmers.se/en/about-chalmers/organisation-and-governance/equality/genie-gender-initiative-for-excellence/

[2] https://reuseit.se/

[3] https://www.ahafestival.se/

poorly understood. In the context of live coding and learning, curiosity may be facilitated by "mutual surprises" that emerge during the "dialogic liveness" of this conversational practice (Blackwell et al. 2022, 176).

As this was the first time we offered the course, we were unclear about several practical aspects, such as whether we should split the 16 girls into groups, use speakers or headphones, and more. Our focus was to act as "guides on the side" and not as the "sage on the stage" (King 1993), an approach that focuses on student-centered learning and moves away from a teacher-centered approach. We used no slides, provided no lecture notes, and assigned no homework. We mainly used hands-on typing on Strudel's text editor to combine theory and practice. We sometimes used the blackboard to make lists of commands, and we only advised them to document the code and practice at home. Furthermore, we ensured to have a secure and safe environment for the students (Maric and Murali Rani 2024a).

2.1.2 Reflective diaries

My main methodological tool for this article is my reflective diaries, which were carried out in an unstructured manner throughout the course. Typically, I was writing my reflections the day after the teaching sessions, but on several occasions, that was done many days after the session or performance. The first diary is before the first day of CC, documenting the course preparations. I wrote six reflective diaries until the first performance on the 2nd of June 2023 and three more diaries until the second performance on the 29th of November 2023. In total, 9 reflective diaries of 2500 words were used as main research material. This material was accompanied by a git repository we used with the second teacher during the development and the course realization.

In pedagogy, reflective diaries are seen as a useful method for the teachers. They have been seen as a method for assessing self-regulated learning (Boekaerts 1999; Wallin and Adawi 2018), for all kinds of learners, either students or educators. Before and during the workshops, I was in active communication with the online community of Strudel, in the Discord server and the GitHub repository. The developers of Strudel and the community provided me with rich insights into the very workings of the language, support, and valuable discussions on teaching.

2.2 Teachers' background and project goals

Before the beginning of the course, during the planning period, we aimed to teach the girls how to write simple pieces of code that can be combined to make a musical composition. Learning to live coding was not our focus, but it was certainly something we had in mind and was considered a successful teaching outcome. As a clarification, live coding, for me, is the modification of a running program while it is running. The musical outcome is seen here as the output of the running program. Thus, I reckon any programming action that can modify the musical outcome is live coding.

I have been live coding for over a decade, and I can say that it was helpful to me to observe how other people live code. Throughout the CC course, I think I never used the term live coding with the students, but I was demonstrating live coding examples in the classroom. Within the community of live coders, it is commonplace that there is no school to learn live coding (Nilson 2007). The second teacher, Kelsey Cotton, also a doctoral student, has an academic background in classical music and a broad performance repertoire, including opera singing, electronic music, and improvisation with musical interfaces. My musical profile is that of an autodidact, as I have no formal music education other than a few lessons on two different musical instruments and 15 years of experience with music programming tools for algorithmic composition and development of musical interfaces.

3 Notes on teaching

3.1 Teaching planning and preparation

It was the first time I was preparing teaching material for school students, and the same applies to the second teacher. Our initial plan was to do four weeks of teaching (weeks 1-4) and for the remaining weeks (weeks 5-10) to share code and sounds in the classroom as preparation for the concert.

Based on previous research in live coding and pedagogy (Aaron 2016; Aaron, Blackwell, and Burnard 2016), we initially aimed to use Sonic Pi[4] throughout the course. Later on, we got a donation of 16 Chromebooks, and we decided to rely on web-based tools because of ChromeOS contraints. During the course planning, Strudel was a project that was a few months old, and none of us had experience with the tool. After online research and questions on the Discord server for Tidal/Strudel, we realized that Strudel could be an excellent decision. Although this decision sounded risky at that time, we decided it was worth the risk of failure – an inherent quality of live coding. It is not uncommon that a programming

[4] https://sonic-pi.net/

language in early stage of development can significantly change its syntax and overall design, although that was not the case for Strudel, likely because is already based on a mature language for live coding (i.e., TidalCycles).

The teaching material was prepared and organized on a private GitLab repository to ensure version control. Initially, we discussed whether to introduce the young girls to code editors and version control systems to cultivate students' computer literacy, but we soon realized this task was challenging and maybe out of scope. We decided to proceed with teaching a combination of musical and programming concepts specific to Strudel and broader sound synthesis concepts (e.g., sound filters). Besides Strudel, we also introduced the girls to the Freesound database[5], where we imported online sound samples into Strudel.

3.2 Setting the ground for teaching

Out of the 16 girls who started the course, 8 girls dropped out during the first half. The reasons for this large number of dropouts are presented by Maric and Murali (Maric and Murali Rani 2024b), and they are multi-faceted. The research methodology employed both quantitative and qualitative data, but due to the large dropout rate and the small sampling size, the quantitative data were not statistically significant (Maric and Murali Rani 2024a, 2024b). Quantitative data were collected using the Likert scale with questions like "How many stars would you give for art" and "How many stars would you give for coding". Qualitative data were collected using open-ended focus group interviews. Also, the research assistant observed all teaching sessions and wrote notes and diaries. The weekly diary aimed to monitor any transitions and progress during the course (Maric and Murali Rani 2024a).

As the CC course has three pillars of girls empowerment, building of social capital and cultural capital, the participants presented a unique case a rich mixture of sociocultural backgrounds. As such, the reasons for high dropout rate can vary from social bonding, aspects of intrinsic motivation to the educational material, self-esteem, cultural heritage, and aspects related to teaching (Maric and Murali Rani 2024b). Indicatively, cultural heritage aspects had an important influence as all the remaining 8 girls were of Indian and Chinese descent.

The remaining 8 girls were the most young, most of them between 10 to 12 years old. Some girls, especially the younger ones, faced considerable challenges during typing, and a similar observation was reported in (Corvi, Mori, and Nulli 2023). For example, to connect to the university Wifi, the students had to type a long password (more than 10 characters long and unreadable) within a no-preview text field. That was a significant challenge for the younger girls, which extended to other aspects, such as difficulties in finding special characters on the keyboard (i.e., characters that require a combination of key presses, thus including the complexity of the typing interface). Another example is that the old Strudel URL https://strudel.tidalcycles.org was somewhat challenging to type, which is no longer the case given the new URL https://strudel.cc. Another observation is that this generation of students is not familiar with URLs, maybe because of mobile apps.

Some of the girls had musical training, but many did not have any previous training. The same applies to the programming experience of the young girls. None had previous experience with textual programming (Maric and Murali Rani 2024a), but only a few girls had experience with visual-based programming on Scratch (Resnick et al. 2009).

3.3 Routine in the classroom

The initial plan was to create four groups out of four students each. In practice, this never worked out, for various reasons, from age difference, student dropouts, absence during the sessions, and more. During the first workshops, we decided to split the 16 girls into two age groups, as we noticed significant developmental differences between older and younger girls. When half of the group dropped out from the workshops, we worked in a single group of 8 students, two teachers, and one to two mentors on average per workshop. The senior researcher and the research assistant were also present in the sessions, doing observations and actively engaging with the young girls. Most of the mentors were also programmers, who also helped with practical questions during the workshops.

At the beginning of each session the girls required 15-20 minutes to set up the equipment, which was stored locally in the university building. This duration includes connecting to the WiFi, connecting the soundcard and headphones to the computer, and then connecting each soundcard to a sound mixer and a stereo loudspeaker system. At the end of each workshop, we allocated time to play their Strudel composition on the loudspeakers to experience how they may sound in a performance setting (Figure 1).

[5] https://freesound.org/

Figure 1: *A photograph from the classroom.*

3.3.1 Strudel specifics over the workshops

The workshop began with us typing on Strudel's web interface by explaining the interface and introducing new commands (i.e., "play," "share," buttons on the top of the browser, and "sounds," "reference" menu at the bottom). We introduced the notion of patterns and explained how to use double quotes and how to comment on different lines of code. During the first and second workshops, we introduced the girls to use the commands `sound` and `note` to play drum sounds and melodies correspondingly. We also demonstrated how to load different sound samples using the sound banks. This worked well, and the girls understood quickly that they had to use the command `stack` to overlay different sound patterns. We use the Latin alphabet names of the musical notes (e.g., `note("a b d")`), which also worked well in comparison to the MIDI numbering codes and introduced the musical pause (~). Using sharps, flats, and octaves was also easy for them, although the flats may be somewhat confusing because of the character `b` following the note character (e.g., `note("a bb c#”)`, where `bb` is B-flat).

For this reason, a good practice is to make explicit in teaching to use uppercase letters for notes and lowercase for flats (i.e., `note("A Bb C#")`). During the second workshop, one of the girls found a bug in the Strudel notation for note/event duration, which she debugged by herself! That was a parsing error caused by an empty space between the duration character and the closing double quote of the musical pattern (e.g., `note("a b@2 ")`)[6].

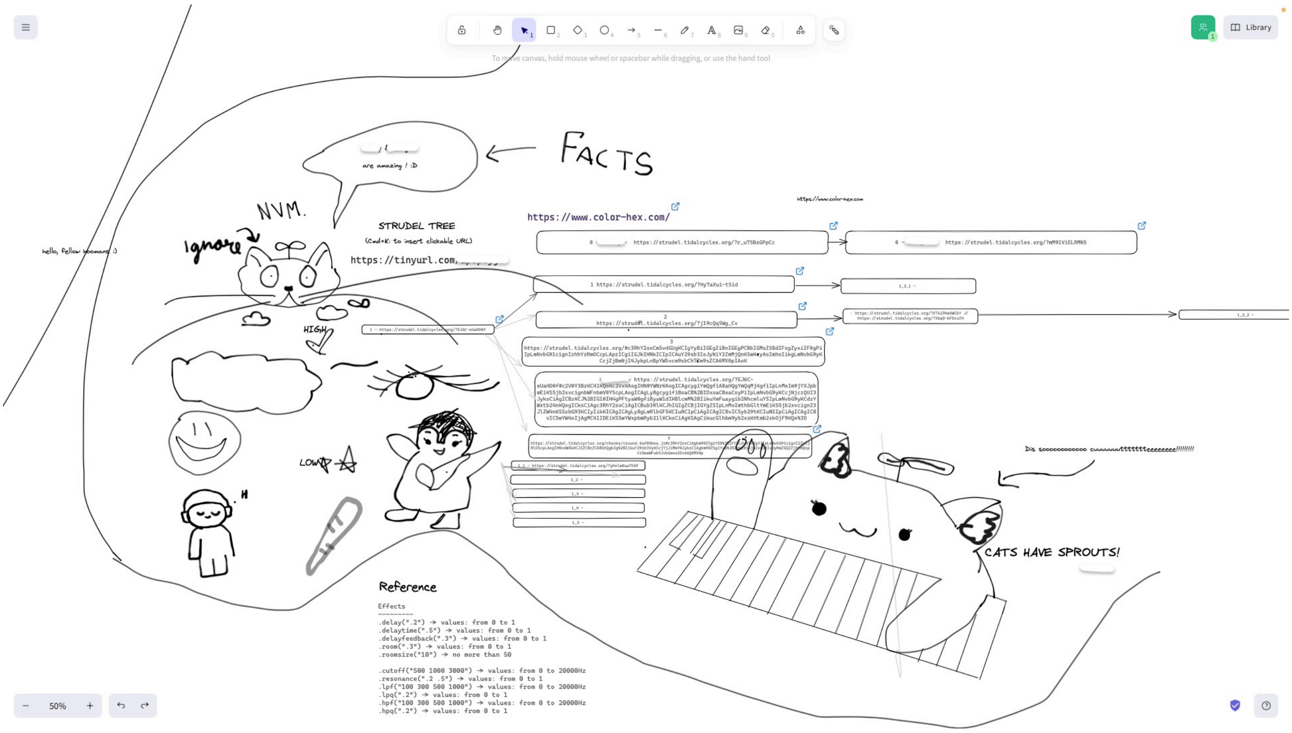
During the third and fourth workshops, we had introduced volume control (`gain`, `velocity`), event multiplier (`ply` and `*`), speed and temporal modifiers (`fast`, `slow`, `early`, `late`), tempo control (`setcps`, `cpm`), grouping events (i.e., chords `[a, b]` and temporal grouping `[a b]`), alternating events (<>) and pianoroll visualizations (`pianoroll`) and code visualizations using colors (`color`). Most of the abovementioned commands were easy for the students and they use them fluently throughout the course. One command that I found was easy to understand but not used by the students was the `speed` command, which stretches the sound samples by a factor.

An overall note is that for certain commands often used in Strudel, such as making sequences using double quotes (`""`), using square brackets for grouping (`[]`), and using alligators clips (<>) for alternating events, we did not show the equivalent human-readable command, such as `seq`, `cat`. One exemption that the `seq` command became necessary to show was one student that had a long sequence of notes, maybe more than a hundred, and had become difficult to monitor. In this case, we showed the `seq` alternative, and the student broke down the long sequence of notes into multiple lines of code.

During the fourth workshop, we planned to introduce sound effects (`room`, `delay`, `delaytime`) and filters (`cutoff`, `resonance`), panning (`pan`, `jux`), along with the conceptual abstraction of sound frequencies (Hertz as unit measurement). We also introduced them to aleatoric elements (i.e., `rand` command). Whereas the girls could hear the difference between a sound with no reverberation and a sound with high reverberation time, we found that sound effects, panning, and filters were rarely used during their composition/performance.

We showed more commands, but I will not get into details as it may be impossible to document every command. All these were quite confusing for the girls, and I can verify that none of these concepts was used in their final compositions/performances. All this showed me that the girls were frugal with their tools, and they could produce complex and aesthetically pleasing music using a small battery of basic commands. Here, I am advocating how advanced levels of complexity can be achieved using simple tools.

[6] GitHub issue: https://github.com/tidalcycles/strudel/issues/546



FACTS
[redacted] are amazing ! :D
NVM.
Ignore
STRUDEL TREE
(Cmd+K to insert clickable URL)
https://tinyurl.com/
hello, fellow [redacted] :)
HIGH
LOW
https://www.color-hex.com/
https://www.color-hex.com
0 - [redacted] - https://strudel.tidalcycles.org/?r_u7SBsGPpCc
0 - [redacted] - https://strudel.tidalcycles.org/?mM91ViELRMN5
1 https://strudel.tidalcycles.org/?Hy7aXui-tSid
1_2_1 -
2 https://strudel.tidalcycles.org/?j19cQqSWg_Cv
: https://strudel.tidalcycles.org/?XTk1PHwXWCHY // https://strudel.tidalcycles.org/?XbqD-kFDczZ4
1_1 - https://strudel.tidalcycles.org/?yHolmbu
1_2 -
1_3 -
1_4 -
1_5 -
Dis sooooooooooo cuuuuuuuutttttttteeeeeeee!!!!!!!!!
CATS HAVE SPROUTS!
50%
Library

Reference

```
Effects
---------
.delay(".2") → values: from 0 to 1
.delaytime(".5") → values: from 0 to 1
.delayfeedback(".3") → values: from 0 to 1
.room(".3") → values: from 0 to 1
.roomsize("10") → no more than 50

.cutoff("500 1000 3000") → values: from 0 to 20000Hz
.resonance(".2 .5") → values: from 0 to 1
.lpf("100 300 500 1000") → values: from 0 to 20000Hz
.lpq(".2") → values: from 0 to 1
.hpf("100 300 500 1000") → values: from 0 to 20000Hz
.hpq(".2") → values: from 0 to 1
```

Figure 2: *The outcome of the first ping-pong game is shown on the collaborative whiteboard.*

Our initial plan to do four weeks of teaching Strudel and six weeks of concert preparations drifted a little, and during the fifth workshop, we introduced the girls to sound recording and Freesound. We showed them how to import sound samples from Freesound to Strudel. Here, it is worth mentioning that this is not an easy process, as it requires to access the HTML source code. That was confusing for the girls initially, but I think they eventually figured out how to deal with this task. The sound recording session was fun for all of us, but the important thing is that the girls understood the notion of sound sampling and semantic browsing of sounds using keywords and how to use sound samples in their songs.

3.3.2 Concert preparations

From the sixth week onwards, we focused on the concert preparations. We discussed group and solo performances and provided more *speaker time* at the end of each workshop. The girls worked in groups or individually on their compositions during these weeks. Each student/group had a different pace, and during week 10, we conducted a dress rehearsal during the workshop.

While helping the students develop their musical compositions, we experimented with various approaches. During the same period, Strudel's Discord server had a ping-pong game using Strudel short URLs. The *ping-pong game* was a collective and turn-based practice of making musical variations of different Strudel scripts. I discussed this practice with my teacher colleague, and we decided to try it in the classroom. We set a maximum of five variations. We used an open-source web application implementing a collaborative whiteboard tool https://excalidraw.com for the ping-pong game and tried it in one of the workshops. The end result is shown in Figure 2. It is obvious that the young girls enjoyed experimenting with the collaborative whiteboard tool, to the point that it was difficult to access the online document as the web interface was overloaded with graphical elements. After this experience, we used an online Word document, and we motivated the girls to work on variations of their collaborative or solo compositions.

3.4 What worked well and what did not

3.4.1 Simple concepts and group activities

The girls enjoyed our group activities, such as the recording session and the *ping-pong game*. Although the end product of the first ping-pong game using the collaborative whiteboard was not productive, that was likely due to the fun drawing activities the girls did on the whiteboard application. The iterative variations worked well when we decided to use an online Word document.

Furthermore, the students appreciated the simple programming and musical concepts and managed to do complex musical compositions using a limited collection of Strudel commands. To summarise, narrowing down to a small battery of programming commands, such as those listed in Table 1, can be a good start for any workshop. It is important here to consider that the students had to be also informed about several more details like the names of the built-in drum sounds (e.g., `bd`, `sd`, `hh`), the musical notes Latin characters, when to use single or double quotes, what are the numerical parameters for the different commands and more.

Command	Alternative	Function
`note`	`n`	Play notes
`sound`	`s`	Play sound samples
`stack`	-	Play items at the same time
`gain`	-	Gain of an event
`velocity`	-	Velocity of an event
`slow`	-	Slow down a pattern by a factor
`fast`	-	Speed up a pattern by a factor
`setcps`	-	Control the global tempo
`cpm`	-	Control the tempo per pattern
`color`	-	Adjust the color of an event
`pianoroll`	-	Display the piano roll
—————-	—————-	——————————————
`””`	`seq`	Create a musical pattern
`,`	-	Make chord and simultaneous sounds
`[]`	-	Group events in time
`<>`	-	Alternate events in time
`*`	`ply`	Make multiple events
`@`	-	Event duration

Table 1: Small battery of Strudel commands commonly used by the students. The second column shows equivalent alternatives taught during the workshops.

3.4.2 The importance of visualization

The visual elements are certainly one of Strudel's strongest assets. The code highlighting in Strudel[7] is a highly educative element, and the `pianoroll` visualization makes the experimentation with Strudel highly attractive to young students. Also, the young girls showed increased interest in color visualizations, as was reported by Maric and Murali Rani (2024a). When emojis were made available in Strudel (that was after the first performance in June 2023), the girls showed increased enthusiasm. Also, the interactivity of the sliders[8] was something easy to grasp but maybe hard to implement in practice, possibly due to nested syntax and range parameters, but also because the feature was introduced after the first concert, so the girls had only a few opportunities to familiarize themselves with sliders. All in all, experimentation with the visual elements of the language can offer many opportunities to Strudel and live coding as a research field and performance practice. The recent development of running Hydra within Strudel[9] is a fascinating development to this end, and there are numerous more possibilities with more JavaScript-based programming environments.

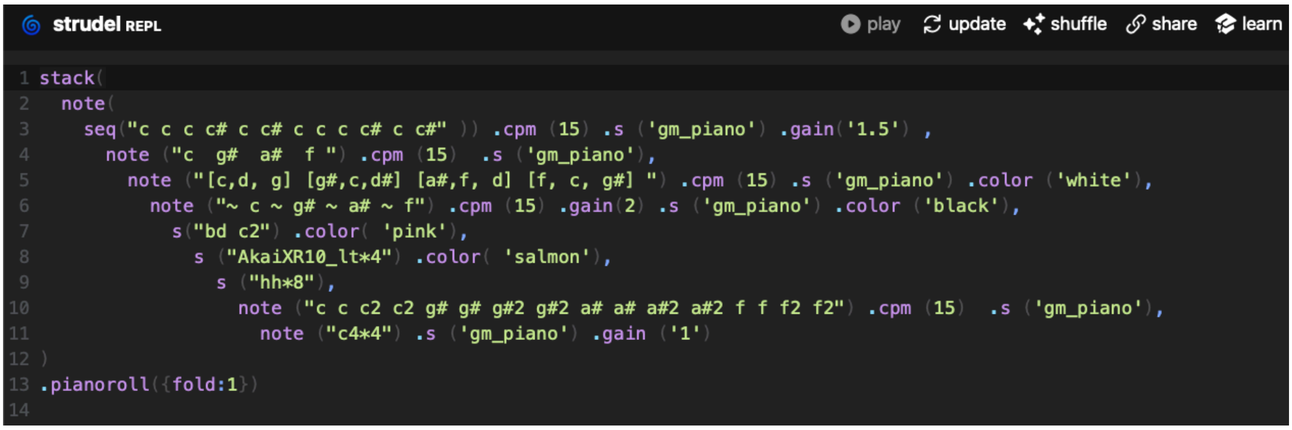
3.4.3 Code readability and formatting

On the programming side, I noticed that younger girls had difficulties with code formatting. Common mistakes were often related to nested parentheses. An interesting indentation style is shown in Figure 5. In this case, the student has made a rather unusual, progressive indentation. This is likely because the girl commented on uncommented lines of code during her performance to modify the musical structure. It may be that this type of progressive indentation can visually help the coder during such practices. It is also interesting to note the empty spaces between the parentheses and the Strudel commands.

[7] Likely based on Alex McLean's experimental text editor for TidalCycles called feedforward https://github.com/yaxu/feedforward.

[8] An example using a slider: https://strudel.tidalcycles.org/?yCaBBuyyJTkM

[9] https://github.com/atfornes/Hydra-strudel-extension



```js
stack(
  note(
    seq("c c c c# c c# c c c c# c c#" )) .cpm (15) .s ('gm_piano') .gain('1.5') ,
      note ("c  g#  a#  f ") .cpm (15)  .s ('gm_piano'),
        note ("[c,d, g] [g#,c,d#] [a#,f, d] [f, c, g#] ") .cpm (15) .s ('gm_piano') .color ('white'),
          note ("~ c ~ g# ~ a# ~ f") .cpm (15) .gain(2) .s ('gm_piano') .color ('black'),
            s("bd c2") .color( 'pink'),
              s ("AkaiXR10_lt*4") .color( 'salmon'),
                s ("hh*8"),
                  note ("c c c2 c2 g# g# g#2 g#2 a# a# a#2 a#2 f f f2 f2") .cpm (15)  .s ('gm_piano'),
                    note ("c4*4") .s ('gm_piano') .gain ('1')
)
.pianoroll({fold:1})
```

Figure 3: *A musical composition from the concert on Strudel shows an unusual and progressive code indentation style.*

3.4.4 Advanced sound synthesis concepts

Filters, sound effects, and panning are three things that the girls knew about, but they did not use them in practice. It may be that such concepts require an advanced understanding of sound production and music-making, and I can verify from my own practice that several times I forget panning during my live coding practice. Furthermore, more abstract concepts, such as envelopes and audio frequencies, were hard to conceive, something also understandable as I remember as an undergraduate student, I was struggling to understand what is the power spectrum.

4 Concerts and performance practices

During the first performance, there were 8 performance acts and 11 compositions in total (Figure 4). There were four duo performances using a single laptop, and four solo performances. All performances were rehearsed during the dress rehearsal, where we used the same laptop for all the girls. We decided to do the same for the concert, and on stage, we had one computer and a document with the Strudel short URLs of the girls' compositions. In this manner, we could avoid any troubleshooting on stage. The duration of the girls' performance act was ranging from one to four minutes. Some of the girls requested a signal to finish the performance, others used a timer.

We simulated a concert scenario during the dress rehearsal so that the girls performed without interruption. The process was that we used an online document with the Strudel short URLs, and each performance act was next to another. One of the girls had commented out most of the parts of the code. When it was her turn to rehearse, she unintentionally pressed the play button on the Strudel interface. When that happened during the dress rehearsal, the other teacher and I motivated her to uncomment line-by-line and update the script on the fly. That was a practice the girl was already doing on her laptop when preparing her compositions, I would guess as a matter of simplicity to break down the complexity and listen to each pattern individually. What was unclear to the girl was whether uncommenting and updating the script on-the-fly would have artistic value and, I would guess, not merely be perceived as a "childish" experimentation on stage.

After the end of the dress rehearsal, we motivated the above-mentioned girl to practice at home and perform in a similar manner during the concert. What is surprising is that during the concert proper, another duo performed in a similar manner (i.e., by commenting and uncommenting patterns), and we were not aware at all of it (i.e., the teaching team). Another case of commenting and uncommenting a line of code, was to play a musical pattern only once, similar to how the TidalCycles function `once`, which is not yet implemented in Strudel.

During the second concert on November 2023, five solo acts were conducted in total. During that concert, another girl established a similar performance practice. During the dress rehearsal before the second concert, I motivated her to write a score for her performance in a manner of setting different durations and commenting/uncommenting lines on code. The example I provided her during the dress rehearsal is shown below, where `.add` and `.slow` indicate Strudel functions in the performance script.

~~~~ {js} 1. first start with line #2 (.add) commented — a few seconds 2. then you uncomment line #2 (.add) — a few seconds 3. then you comment line #5 (.slow) — a few seconds 4. then you comment line #2 (.add) — a few seconds ~~~~
~~~~

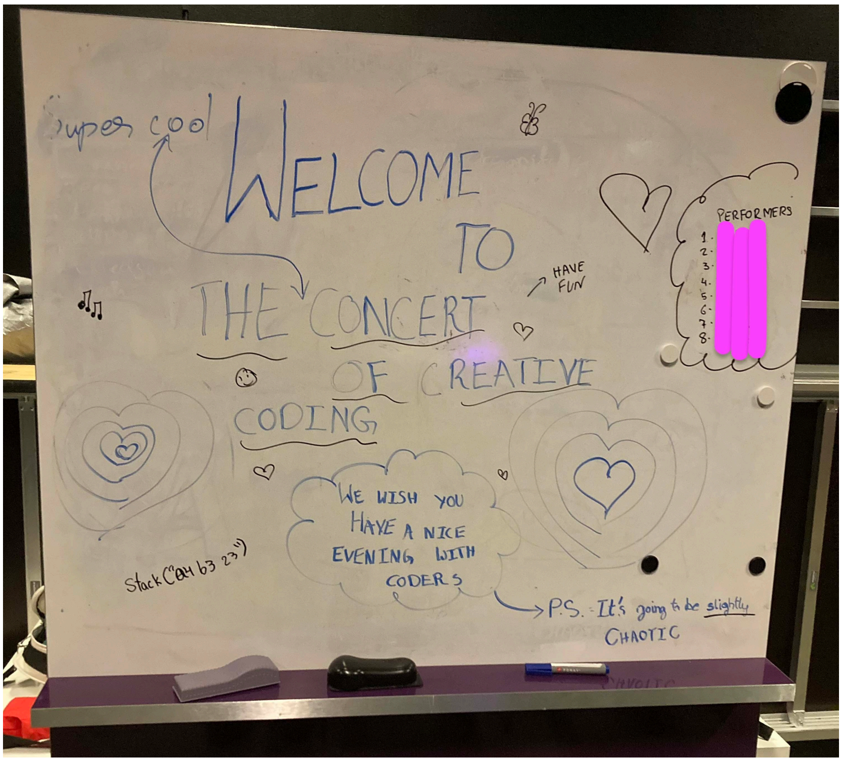
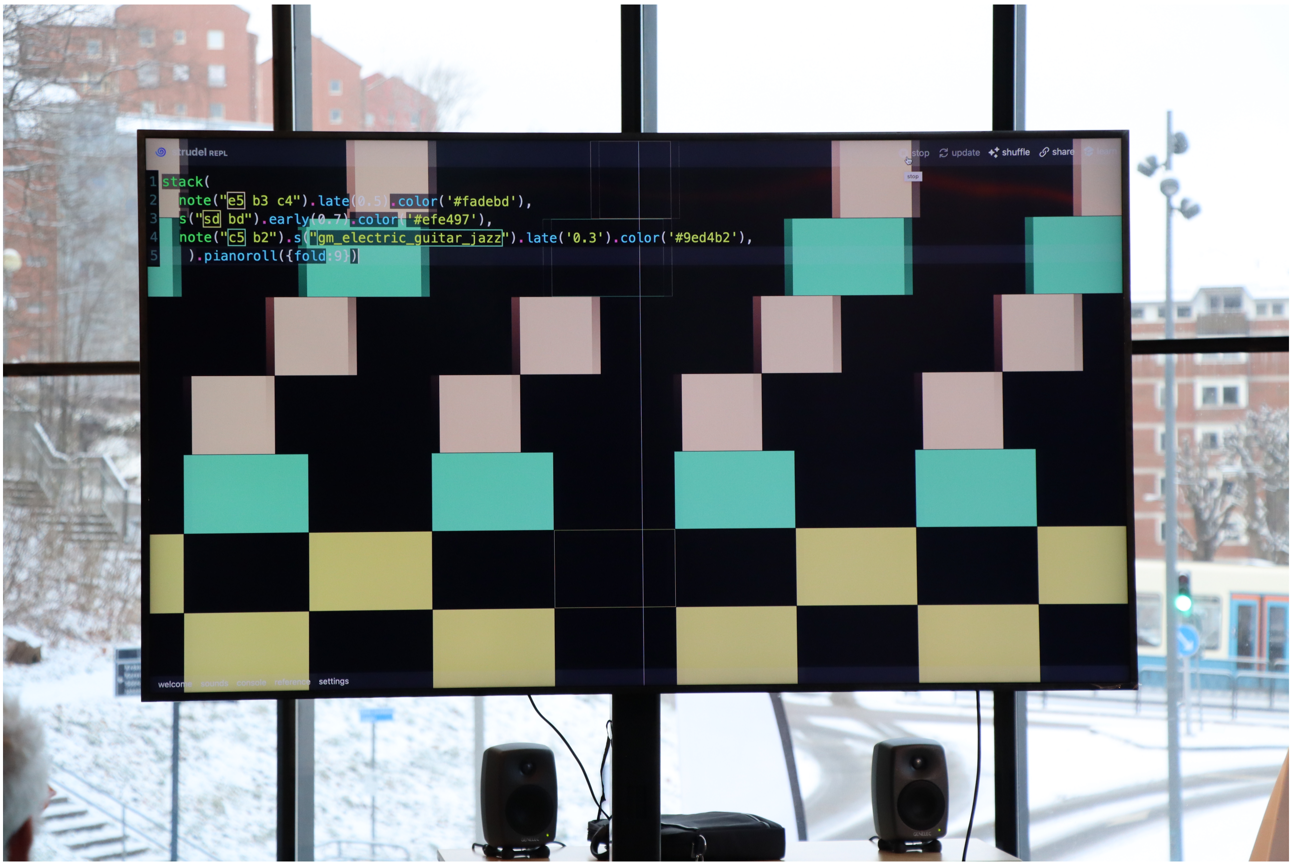

Super cool
WELCOME TO THE CONCERT OF CREATIVE CODING
HAVE FUN
PERFORMERS
1. 2. 3. 4. 5. 6. 7. 8.
[redacted]
WE WISH YOU HAVE A NICE EVENING WITH CODERS
Stack("a4 b3 23")
P.S. It's going to be slightly CHAOTIC

Figure 4: *Girls drawings on a whiteboard in the concert room.*



```
stack(
  note("e5 b3 c4").late(0.5).color('#fadebd'),
  s("sd bd").early(0.7).color('#efe497'),
  note("c5 b2").s("gm_electric_guitar_jazz").late('0.3').color('#9ed4b2'),
   ).pianoroll({fold:9})
```

Figure 5: *A photograph from the second concert during the AHA festival.*

5 Discussion

Previous research on musical live coding and education indicates several characteristics of effective teachers (Burnard et al. 2017), such as mastering the art of unlearning, combining theory with practice, learning from observation, using exploratory tools, and using structured peer support. When we started the CC course, we were unfamiliar with Strudel. In my artistic live coding sessions, I use SuperCollider (SC3) programming language (McCartney 2002), and while I had a little experience with TidalCycles, I can say that using Strudel is nothing like SC3. In Strudel, I can generate sound within a few seconds when starting a session from scratch, whereas in SC3, that would take one to two minutes. The numerous hours I spent practicing Strudel and my background in live coding enabled me to form a basic theoretical understanding of Strudel to communicate this knowledge to the girls effectively. Strudel is a highly exploratory tool, and the pattern implementation inherited by TidalCycles may be impossible to understand on a theoretical level, so practicing is inevitable. To this end, the visualization functions of Strudel help enormously the user.

In Section 4, I discussed how the girls get started to live code both when doing and simulating a performance. In essence, this is how the girls realized the notion of *liveness*, as they combined "performing and experimentating simultaneously" (Burnard et al. 2017) (p. 2). This is the main point I would like to make by writing this article, and I can say it was a highly surprising and highly rewarding outcome for us as teachers.

Another occasion when live coding became evident to the girls was when they would like to play a musical pattern only once without repeating it repeatedly. That occurred for a couple of students who wanted to do an intro for their compositions. While this functionality is already implemented in TidalCycles with the function `once`, in Strudel there is no equivalent function.

From a different angle, the importance of visualization in teaching live coding is one of the main arguments of this article. Strudel developers have conducted an enormous effort into developing various sorts of visual elements, such as sound event highlighting, unreadable fonts that look like asemic writing[10], spiral piano roll visualizations, and emoji visualizations, among others. The built-in visualization feature of highlighting musical events is of particular importance for the students to make a mental model between the written code and the musical outcome. When we tried to use flok.cc with two of the girls to conduct a duo performance, we encountered no sound event visualization, which was confusing for the girls. How an equivalent feature would look when making sound synthesis with oscillators is yet to be shown, but Gibber is one of the tools that leads in this aspect (Roberts et al. 2022). One proposal that comes to mind based on the girls' practices and Strudel's interface is that it may be beneficial to modify the piano roll and text visualizations when applying sound filters and sound effects. For instance, the command `gain` varies the brightness of the sound events on the piano roll. Maybe a similar approach could be useful for education in the case of `cutoff` or `room`. An example could be to modify the texture of colored events.

On the specifics of the girls' performances, my observations are in agreement with the study by Burnard et al. (2017). Some of the girls presented curatorial compositions. For instance, one of the girls used a ready-made composition available on the Strudel web interface when pressing the "shuffle" button and modified the colors. Some girls conducted arrangements on Strudel's ready-made examples or code examples provided in the classroom, whereas others composed their pieces from scratch (Burnard et al. 2017). I noticed that it was really popular in the classroom to use a web application that simulates a piano, using either the mouse or the keyboard to play the notes. I found out that this piano simulation software was mostly used by girls who had musical training.

A thematic analysis that was conducted on observations during the course by the senior researcher and her research assistant (Maric and Murali Rani 2024a) shows that the girls were able to understand the syntax and commands of Strudel. The same report identifies the difficulties of the girls in remembering a wide range of commands in Strudel, especially for students younger than 12 years old (ibid.). Furthermore, the thematic analysis showed that the students were experimenting with sound and visual aspects of Strudel in an iterative manner, which is in agreement with Freeman and Magerko (2016).

In teaching, a lack of clarity in project objectives was also an important factor in high dropout rates (Maric and Murali Rani 2024b). It was often the case that we, as teachers, did not assign clear roles to the students. One important aspect was that we did not manage to form project groups. A group scenario would have enabled us to easily assign clear roles and tasks, such as student A working on a drum pattern, student B working on a melodic pattern, and so on. This uncertainty of clear tasks and goals contributed to group disengagement and a lack of clear goals. There is essentially a chain reaction between different aspects that range from cultural to teaching aspects, and can contribute to building a sense of perceived difficulty and can result in decreased amounts of behavioral, emotional, and cognitive engagement with the activity. These are some reasons for the high rate of dropouts. Maric and Murali Rani (2024b) provide a thorough list of interventions to mitigate the high dropout rate, such as culturally responsive teaching, tailoring teaching

[10] Asemic writing is a form of writing with no semantic meaning. In the case of Strudel, the unreadable fonts exhibit code semantics, which raises a question of how live coding is even possible when one cannot read the written code. From my practice, experimenting with the non-readable fonts in Strudel is a fun and challenging activity, as I know what I am writing and listening to it, but it becomes difficult to go back and edit – maybe we can call it "one-hot live coding" as what is typed is likely set in stone, unless the coder seeks for radical experimentation and potential system failure.

strategies to age groups, tailoring learning environments, and more. The reasons that older girls drop out may be related to emotional and cognitive engagement, such as lack of interest, negative moods, and internalizing behaviors (ibid.).

For future iterations of the Creative Coding course, it may be helpful to the students to create visualizations between programming concepts and real-world examples but also metaphorical examples. For instance, a real-world example would simply be to show a drums set and point out the `bd`, `sd`, and so on. A metaphorical example might be to use a real sieve with sand and larger grain sizes to demonstrate how the `cutoff` command works. Further experimentation with spinning wheels or dice containing Strudel commands or parameters may be useful to introduce the student to aleatoric elements. Regarding collaborative live coding sessions, a straight-forward case would be to share a laptop among the performers, may be called a 'hot-seat' scenario, similar to how a relay race performance may look like[11].

Finally, a design recommendation for the Strudel REPL or other programming languages focusing on the education of programming would be to implement a 'kids' template that includes a reduced set of features. For instance, Strudel's "reference" page includes over a hundred commands. A 'kids' template may be designed to include only a small subset of the actual commands (Table 1). The same logic can be applied to the sound samples or the interface design as a whole.

6 Conclusion

The study focuses on musical and programming aspects of teaching Strudel to young girls using my self-reflections as main methodological research material. I present detailed descriptions of practical and logistical things that worked well, and teaching practices that are either a no-go for the specific age group, or inefficient.

The key point of this article suggests that the girls realized musical live coding during performance practice, while not being aware of live coding as a concept. Live coding rather became a necessity during the dress rehearsal for one of the students, and during the two concerts more students employed similar performance practices.

I suggest that teaching and learning Strudel for this specific age group, of 10-15 years old, can begin with a small battery of programming commands, no more than 20 different commands. For both concerts, the girls used a small vocabulary of Strudel commands, demonstrating that they were frugal with their tools. Further suggestions for advancing visualization elements of the specific tool are provided. The significance of the findings demonstrates how Strudel can be an excellent tool for teaching music-making using text-based programming. The study is limited by its subjectivity in terms of methodological design and the large amount of student dropouts.

References

10 Aaron, Sam. 2016. "Sonic Pi–Performance in Education, Technology and Art." *International Journal of Performance Arts and Digital Media* 12 (2): 171–78.

Aaron, Sam, Alan F Blackwell, and Pamela Burnard. 2016. "The Development of Sonic Pi and Its Use in Educational Partnerships: Co-Creating Pedagogies for Learning Computer Programming." *Journal of Music, Technology & Education* 9 (1): 75–94.

Blackwell, Alan F, Emma Cocker, Geoff Cox, Alex McLean, and Thor Magnusson. 2022. *Live Coding: A User's Manual.* MIT Press.

Boekaerts, Monique. 1999. "Self-Regulated Learning: Where We Are Today." *International Journal of Educational Research* 31 (6): 445–57.

Burnard, Pamela, Franziska Florack, Alan F Blackwell, Sam Aaron, and Carrie Anne Philbin. 2017. "Learning from Live Coding." In *The Routledge Companion to Music, Technology, and Education*, 61–72. Routledge.

Corvi, Francesco, Giovanni Mori, and Giovanni Nulli. 2023. *Live Coding and Education. A Practical Experience.* Zenodo. https://doi.org/10.5281/zenodo.7843819.

Freeman, Jason, and Brian Magerko. 2016. "Iterative Composition, Coding and Pedagogy: A Case Study in Live Coding with EarSketch." *Journal of Music, Technology & Education* 9 (1): 57–74.

Freeman, Jason, Brian Magerko, Doug Edwards, Tom Mcklin, Taneisha Lee, and Roxanne Moore. 2019. "EarSketch: Engaging Broad Populations in Computing Through Music." *Communications of the ACM* 62 (9): 78–85.

Kereluik, Kristen, Punya Mishra, Chris Fahnoe, and Laura Terry. 2013. "What Knowledge Is of Most Worth: Teacher Knowledge for 21st Century Learning." *Journal of Digital Learning in Teacher Education* 29 (4): 127–40.

[11] Sieve algebra study for groups of unknown size by Dennis Scheiba and Julian Rohrhuber: https://www.youtube.com/live/e4LFbfsqPv0?si=JMc-wLqhnRnMmzUG&t=19848

King, Alison. 1993. "From Sage on the Stage to Guide on the Side." *College Teaching* 41 (1): 30–35.

Maric, Jasmina, and Lekshmi Murali Rani. 2024a. "Singing Code." *(Under Submission)*.

———. 2024b. "The Wicked Problem of Dropouts." *ACM CHI Conference on Human Factors in Computing Systems 2024 (Accepted)*.

McCartney, James. 2002. "Rethinking the Computer Music Language: Super Collider." *Computer Music Journal* 26 (4): 61–68.

McLean, Alex. 2009. "Embodied Creativity." In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

McLean, Alex, and Geraint Wiggins. 2010. "Tidal–Pattern Language for the Live Coding of Music." In *Proceedings of the 7th Sound and Music Computing Conference*, 331–34.

———. 2012. "Computer Programming in the Creative Arts." In *Computers and Creativity*, 235–52. Springer.

Nilson, Click. 2007. "Live Coding Practice." In *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, 112–17.

Resnick, Mitchel, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, et al. 2009. "Scratch: Programming for All." *Communications of the ACM* 52 (11): 60–67.

Roberts, Charlie, Ian Hattwick, Eric Sheffield, and Gillian Smith. 2022. "Rethinking Networked Collaboration in the Live Coding Environment Gibber."

Roos, Felix, and Alex McLean. 2023. *Strudel: Live Coding Patterns on the Web*. Zenodo. https://doi.org/10.5281/zenodo.7842142.

Rubin, Marc J. 2013. "The Effectiveness of Live-Coding to Teach Introductory Programming." In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 651–56.

Ten, Alexandr, Pramod Kaushik, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. 2021. "Humans Monitor Learning Progress in Curiosity-Driven Exploration." *Nature Communications* 12 (1): 5972.

Wallin, Patric, and Tom Adawi. 2018. "The Reflective Diary as a Method for the Formative Assessment of Self-Regulated Learning." *European Journal of Engineering Education* 43 (4): 507–21.

Ward, Adrian, Julian Rohrhuber, Fredrik Olofsson, Alex McLean, Dave Griffiths, Nick Collins, and Amy Alexander. 2004. "Live Algorithm Programming and a Temporary Organisation for Its Promotion." In *Proceedings of the README Software Art Conference*, 289:290.