Episode 1: Critical Code

Fri, 6/18 2:29AM • 21:27

**SUMMARY KEYWORDS**

languages, programming languages, programmers, code, people, computer, art, artists, ramsey, designing, create, write, programming, early, english, computation, sarah, media, thinking, hackers

**SPEAKERS**

Grace Hopper (recording), Daniel Temkin, Lee Tusman, Ramsey Nasser, Sarah Groff Hennigh-Palermo

**Lee Tusman**

You're listening to artists and hackers, the show dedicated to the community building new digital tools of creation. We talk with programmers, artists, poets, musicians, bot makers and more. We're looking at the current palette of art making tools online, and take a critical eye to the history of technology and the internet. We're interested in where we've been, and speculative ideas on its future. In this episode, we delve into the community of artists who create their own languages, and I don't mean Klingon, Elvish, or Esperanto. Programming languages are made up of instructions that a programmer uses to produce various kinds of output. These instructions when put together implement algorithms. The first programming languages predate the use of a digital computer. In fact, the earliest might have been the automatic flute player of the ninth century described by the astronomers and brothers Musa from Baghdad during the Islamic golden age and their book of ingenious devices. The earliest modern computers such as codebreaking machines in the Second World War did not operate based on computer languages. Instead, they were programmed by rewiring the machine. Soon after this era came machines that can be programmed in binary code using punch cards, or by flipping switches on and off. In the 1950s the US Navy's Grace Hopper decided that business data customers were uncomfortable with mathematical notation,

**Grace Hopper (recording)**

In 1951 it was becoming increasingly clear ....51, 52, 53, 54... there are a large number of people who love symbols, they were scientists and engineers. There are also a large number of people that hated symbols. And they worked in words. They were the data processors. So we proposed that we will permit people to write in English statements. And we'll write a compiler, which will translate the machine code. Of course I was told we couldn't do that because computers can't understand English words.

**Lee Tusman**

She led a team to specify an English programming language that was soon called FLOW-MATIC and influenced other early languages like COBOL. In fact, it's influenced all languages since then. And in the past 70 years, there has been the birth of 10s of thousands of programming languages, so you might wonder why new ones would still need to be invented. In this episode, we meet artists and hackers that aren't satisfied with the limitations of programming languages. They create their own languages as art forms or languages that themselves make art. They serve new needs, address current times or center themselves, instead of the business, corporate or military needs of computation past. Sara Groff Hennigh-Palermo is a digital and video artist with a hybrid arts practice. She makes art. She's a programmer, and she's developed her own language, La Habra to create her own visuals and code. We caught up with Sarah from her studio in Berlin to talk about her experience as an artist creating languages and how she explains her own interests and work.

**Sarah Groff Hennigh-Palermo**

Yeah, there's a couple of different ways I've described it in the past that I feel like work for me: like what if Roy Lichtenstein and Agnes Martin had a baby who learned how to code. It's like if you had a surfer T and Ray Gun magazine and smashed them all together, and like took them to a Del Taco. That maybe is the look I'm aiming for in my art. I think that insisting on the importance of the aesthetics and focusing on transcendence as a goal is my critical work, doubly so I would say because being a woman of color, a lot of times people are interested in art, if it's about race, because I think it's a way to make art about white people still, right. Like if you make your art about race, then it has to be about like the relationship that you have with white people. And for me saying like I'm not talking about that, I'm talking about other things that are really important to me in this space does feel like a critical stand, even if it's in the absence of overt messaging.

**Lee Tusman**

Sarah has been active in the live coding community for a number of years, particularly in livecode.nyc. And if you're not familiar with live coding, it's a performance medium where visual artists and electronic musicians stand on the stage or beside it, hunched over laptops, typing in code, and that code is both projected in front of you so everyone can see it. And you're also hearing the music or seeing the visuals that's made from that code.

**Sarah Groff Hennigh-Palermo**

The focus of live code is really about making computing apprehensible, comprehensible... for people to make it something that they can grab onto so you can hear the music and see the code that makes the music. You can see the visuals and see the code that makes the visuals. I became part of livecode.nyc and part of Codie (livecoding group) because Kate Siccio who is now Professor at VCU was a professor at IDM at NYU when I was in the program there. And she had done live code in England in Sheffield. And so she sort of brought live code to New York. And once I graduated from school, she and a couple other friends of mine, including Ramsey Nasser were a part of this live code thing. So I first started just going to meetings so that I could see my friends and also yell about compilers. And Kate talked me into performing with her by basically being like, you'd get to write a whole tool and compiler to make visuals. I got sucked into it through the technical question. And I performed once and it blew my mind. I first got into computer languages as an art form and just fell in love with them at SFPC, the School for Poetic Computation. I was in the second full length class. But when I was at that class Ramsay Nasser was one of our professors. And he's also a good friend of mine now. And this is how I originally met him. And he did a class about how to make computer languages. And I really feel like that class was a huge turning point in regards to my relationship with computers. Before that, I had learned to write code and I felt like computers were these like, capricious Gods like they were like the Greek gods, like, they might come down and pretend to be a swan and sucker you into like having children with them. And if you type the right thing, in your text editor, maybe it would do what you wanted, and maybe it wouldn't, and there was no way to reason about it. Beyond that, I just had to appease the computer. What I took from that was I was just like, oh, like computers made by idiots just like me, like it just made it so understandable. All of a sudden, like it demystified it

**Lee Tusman**

Now, one thing to note about Sarah is she's not just thinking through the relatively recent history of modern computing, but she's really contextualizing her work and computers and culture along a much longer historical timeline. As Sarah is active as an artist and live coder, she's committed to exploring form, color and abstraction. But she's also grounded in political principles. And she writes up a manifesto, which she calls Computer Critical Computer Art.

**Sarah Groff Hennigh-Palermo**

So one thing that I say in the seven points for Computer Critical Computer Art is that computers like to pretend that they're free from history. They're not. And let's remind them of this fact. A big case in which people talk about sublimity is the 18th century looking at nature, you know, wrapped up, of course, in colonization, that people want to talk about, like the sublime of the great forests, or like in the 19th century going to the Western US. And that kind of sublimity is really only is also very much about colonialism and people getting to encounter those spaces. When I take the work I make and it works. And I put it up big on a wall and I stare at it, I feel that same sense of losing myself that I do if I look at a Bill Viola piece, or an Agnes Martin piece, or a Barnett Newman piece, and this is how I figured out how to make that feeling happen. So like I have all these critical ideas, and all of those things are true, but fundamentally at its heart, it's like about being able to find that feeling.

**Lee Tusman**

Sara met Ramsey Nasser at the school for poetic computation, also known as SFPC. It's a sort of hybrid school residency and research group that was founded in 2013. In New York.

**Ramsey Nasser**

My name is Ramsey Nasser. I'm a computer scientist and educator and a game designer. And I'm passionate about making fun and useful things.

**Lee Tusman**

Like Sarah, Ramsey is also an artist and a programmer. And he's always found himself probing the boundaries between these media.

**Ramsey Nasser**

I work on games, and I work on interactive experiences. And I work on programming languages. And in some sense, those are very different artifacts. The goals are very different. But to my mind, they come from the same place. And the similarities there are that you're designing experiences for people, you're thinking, whether you're making a game, or an installation, or a programming language, you're thinking very hard about a human being is going to touch the thing that you made with their hands and with their mind. You're thinking about how you're constraining their actions, what actions you're enabling, what actions you're disallowing, what actions, you're encouraging, or discouraging. You're inviting people to think in particular ways and to act in particular ways. So they're very similar to me: a game and a programming language. And it's a lot of the same skills that I that I feel that I deploy, or a programming language that you want people to use and share. It's not something that you just dream up in your head and bang out on laptop. It's something you share with people and get feedback from people and you integrate that feedback back into the thing. So they're very similar in that regard.

**Lee Tusman**

Programmers are like anyone else. They bring all sorts of interests and goals when they're approaching their craft. For Ramsey, he comes from a place where he's questioning the defaults of the field.

**Ramsey Nasser**

I think a lot of people who write software and use programming languages on a daily basis feel this itch, that the tool that they're using is not great or is full of quirks or and limitations. That's not a novel thought to have. I think most programmers have that thought. And I think a lot of my instincts to get into language design sort of come from that.

**Lee Tusman**

In 2012, Ramsay created the programming language Qalb while in residency at eyebeam Art and Technology Center. He describes the language as both an artistic endeavor and a response to Anglophone bias in programming languages. As Ramsay creates languages as provocation and projects of conceptual art, New York based artists Daniel Temkin creates languages that cut to the core of whether humans and computers can really ever speak the same language.

**Ramsey Nasser**

So Qalb is a response to what I described on the website as Anglophone bias. And what I mean by that is every programming language in use on the planet, every programming language in serious use on the planet is based on American English. This is true without exception. And people ask me about like, you know, what about China? What about India. It's all Java and C++. And they're all people whose native language is not English, whose native script is not even Latin, in fact, are required to gain some familiarity with these, with these language systems in order to become proficient programmers. So the jumping off point for me is that sort of political observation. It's an artistic response, because it does not really set out to solve the problem on its own. So it doesn't set out to like offer a solution, but it's sets out to to be a provocation, and to direct attention towards something that is like largely ignored, I think, by most programmers.

**Daniel Temkin**

I'm an artist who works in, I guess, a few different media. I'm very interested in the clash between human rationality and logic, often in the form of computers and our interaction with computers, but but not always. And that's something I explore in net art pieces, designing programming languages, and sometimes in other media, especially photography, which is actually the thing that I studied at my MFA.

**Ramsey Nasser**

Like Sarah and Ramsay, Daniel has a background that starts really beyond programming.

**Daniel Temkin**

I graduated with a BA in philosophy and Communication Arts, and I had to get a job. So I very quickly taught myself very sort of basic web programming, which was really what you needed in 1997, in order to get a job. And so I've been working as a programmer since then, that's my day job. And my conclusion after years and years of programming is that in general code doesn't work.

**Lee Tusman**

Daniel's art practice revolves around creating Esolangs Ramsey's a true polyglot. He can speak Arabic, English, and some Turkish and he can program in dozens of language and has written many of his own.

**Ramsey Nasser**

There's something I find quite joyful about just watching code in one language turned into code in another language. I love going on Google Translate, and seeing what it thinks English words would be an Arabic, or what it thinks Arabic words would be in English. And partially, that's funny to me, because sometimes it completely fails and you get like these really hilarious edge cases. But it's also just what I have been doing in my head for as long as I've had access to language as a human being. I've always had multiple languages just kind of bouncing around in my head. So yeah, translation, just a sort of active constant process of translating back and forth between things. I think it's just a fundamental part of who I am. And that just comes from being you know, born between two cultures and and being bilingual.

**Lee Tusman**

Brainfuck and Befunge are some of the earliest esoteric programming languages, at first a practice mostly practiced by programmers, but over time, artists started to approach and think about these works, including in the late 90s and then into the 2000s in early 2000 and then as a larger community started to work in this field.

**Daniel Temkin**

So basically an Esolang is short for esoteric programming language, and it's the class of languages that are not designed for practical programming experiments and code. Kind of pushing out the limits of what's possible to do in a programming language, challenges to conventional ideas of computation. It's something that really began in the 90s. The first language that's generally accepted as an esolang is INTERCAL, which is from 1972. But the languages that really sort of began the movement of esolangs, both appeared in in 1993, Brainfuck and Befunge. I started designing programming languages for fun, well before then, but it wasn't until I was in art school that I started to recognize that that work was also art. Because before then I thought of it as there's stuff that I do that artists understand, which is photographic. And there's this stuff that other programmers understand which are these esolangs. And I didn't really, I didn't think of them as art because other artists were just confused by the work. It's it seemed like it was too difficult to make that jump into a medium that you can you primarily experience by writing code in it. There's this there's this great essay by Joseph Weizenbaum, who is best known for creating the Eliza chatbot.

**Lee Tusman**

Eliza is an early natural language computer program. And it was created from 1964 to 1966 at the MIT artificial intelligence lab that was run by Joseph Weizenbaum. And it's really a chatbot. It's a it's an early program, that simulates the experience of speaking with a psycho therapist.

**Daniel Temkin**

And he wrote this paper called science and the compulsive programmer, about the kind of cycle of compulsion that happens when you're writing code. So it's like, you write something and it has bugs, and then you fix those bugs. But if you introduce new bugs, and then you fix those things, you get something that that's basically working the way that you want. But there's always something more you can add to it. And it becomes this kind of cycle where you're adding more and more stuff, whether it makes any sense or not. The reason why it's it's frustrating to write code, especially when you're in that cycle, is that there's the sense that you have total control over the computer, it's going to do anything you tell it to. So when you run into problems, it's because you've told it to do the wrong thing. It's like you've set yourself up as your own obstacle.

**Lee Tusman**

So there can be a distinction in languages between those that are structural and playing with form versus those that might have a conceptual, political, social or cultural conceit to their work.

**Daniel Temkin**

There's a book that came out last year by Mark Marino called Critical Code Studies. And it has a chapter on indigenous programming, which has become an area of exploration more widely recently. And there's there's work like Cree# by John Corbett, where it's not just that you're writing code in Cree instead of in English, but also that it brings in the Cree notions of storytelling into the way that you write the programs, so that you're no longer thinking within the metaphors that English provides for programming.

Yorlang is a programming language by Anuoluwapo Karounwi. He's a programmer based in Nigeria.

My name is Anuoluwapo. Currently, I'm a software developer, I've been writing code for about eight years now. And I'm the creator of Yorlang, a programming language that allows you to write code in Yoruba. So making your language is really not something I thought I was going to do. Initially, it was an idea that came from a friend so he was like, oh, it'd be cool if we could write some keywords like for example, the print statement equivalency revised like sọpé. So that's not a bad idea. Let me give this a go. So I set out to write, print HelloWorld, in Yoruba. So that's, that's pretty much it. Really, it was something that came from an idea from a friend jokingly really and then it became something that when I shared it with the public they wanted to contribute to and to play around with.

So as I'm listening to Anuluowapo and Daniel and Ramsey and Sarah talk about their practice of building languages, I'm really considering them within a lineage of designing tools, new tools.

**Daniel Temkin**

So I do think of programming languages as tools and programming languages are tools that come with with a point of view, they're they're designed in such a way that they're set up for you to think in terms of that language in order to achieve your goals. And so it's interesting as art projects, as esolangs, because you're basically designing the point of view that you want people to think through in order to do whatever practical thing they're trying to do with your language. And that's really where the drama happens is, is when people are trying to think through your language in order to achieve some sort of goal with it. In a sense, these are set up as challenges and I think that that's really what makes them interesting.

**Lee Tusman**

Programming languages are a fundamental aspect of computing, and they've expanded far beyond simply acting as tools of mathematics and computation. They have embedded values and artists and programmers are actively shaping them. Today, they're a genuine medium of creative expression, and as artists and hackers bring new esoteric languages into being, their created languages are more reflective of the world that we live in today, as well as works that are playful, expand our understanding, challenge our biases and create new mediums and forms of art and expression.

**Sarah Groff Hennigh-Palermo**

I think it's a real shame and to underestimate what's interesting about computing to just turn them into machines for business and oppression. Like that's crap.

**Lee Tusman**

That's our show this week. You've been listening to artists and hackers. I'm your host Lee Tusman. The show was produced by Mimi Charles. Design and coordination by Caleb Stone. Thanks to everyone we spoke to including Sarah Groff Hennigh- Palermo, Ramsey Nasser, Daniel Temkin, and Anuoluwapo Karounwi. Our music today is Trip to Ganymede by Kielokaz and the songs Serrated and Pixel pool by Codie. Sound designed by Mimi Charles. Support for Artists and Hackers comes from Purchase College. Our website where you can find more episodes, our show notes and additional info at ArtistsandHackers.org. We're on Instagram as ArtistsandHackers and twitter at ArtistsHacking. If you liked our show, please let a friend know. Thanks for listening