EPISODE 975

[INTRODUCTION]

[00:00:00] JM: FreeCodeCamp was started five years ago with the goal of providing free coding education to anyone on the Internet. FreeCodeCamp has become the best place to begin learning how to write software. There are many other places that a software engineer should visit on their educational journey, but freeCodeCamp is the best place to start, because it is free and there are no advertisements.

For most people learning to code, the price of that education is very important, because they're learning to code to build a new career. Many people are learning to code because they don't have any money. It's also important that a new programmer learns from an unbiased source of information, because an ad-supported environment will educate the new programmer towards products that they might not need.

FreeCodeCamp has not been easy to build. Building freeCodeCamp has required expertise in software engineering, business, media and community development. The donation-based business model of freeCodeCamp doesn't collect very much money. Why would somebody build a nonprofit when they could spend their time building a highly-profitable software company?

Quincy Larson is the founder of freeCodeCamp, and he joins the show for a special episode about his back story and the journey to building the best place on the Internet for a new programmer to begin. This was an enjoyable episode for me, because I am friends with Quincy. I've have spent a lot of time with him, but I actually knew nothing about his background. If you think about it, it's pretty interesting that somebody would start a nonprofit with his collection of skills. So why would somebody do that? Well, the answer is in this interview, because he has a very distinctive background with a lot of ups and downs.

Learning about Quincy's story help me understand in much greater detail why he built freeCodeCamp and the related media properties of freeCodeCamp. They have a very popular YouTube channel, a popular podcast, and I recommend checking out those resources.

Thanks to Quincy for coming on the show, and I hope you enjoy this interview with him.

[SPONSOR MESSAGE]

[00:02:25] JM: Apache Cassandra is an open source distributed database that was first created to meet the scalability and availability needs of Facebook, Amazon and Google. In previous episodes of Software Engineering Daily we have covered Cassandra's architecture and its benefits, and we're happy to have Datastax, the largest contributor to the Cassandra Project, since day one as a sponsor of Software Engineering Daily.

Datastax provides Datastax enterprise, a powerful distribution of Cassandra created by the team that has contributed the most to Cassandra. Datastax enterprise enables teams to develop faster, scale further, achieve operational simplicity, ensure enterprise security and run mixed workloads that work with the latest graph, search and analytics technology all running across hybrid and multi-cloud infrastructure.

More than 400 companies, including Cisco, Capital One, and eBay run Datastax to modernize their database infrastructure, improve scalability and security, and deliver on projects such as customer analytics, IoT and e-commerce.

To learn more about Apache Cassandra and Datastax's enterprise, go to datastax.com/sedaily. That's Datastax with an X, D-A-T-A-S-T-A-X, @datastax.com/sedaily.

Thank you to Datastax for being a sponsor of Software Engineering Daily. It's a great honor to have Datastax as a sponsor, and you can go to datastax.com/sedaily to learn more.

[INTERVIEW]

[00:04:05] JM: Quincy Larson, welcome back to Software Engineering Daily.

[00:04:08] QL: Thanks, Jeff. It's an honor and I think it's been about four and a half years since I've been on Software Engineering Daily.

[00:04:14] QL: Well, it’s been a while since you were on to only talk about that. We had some random shows. You were on with Haseeb. We talked net neutrality a little bit. We had some random discussions, but we haven't talked about your magnum opus in a while, which is freeCodeCamp, except for when you come to San Francisco or when you're in San Francisco living here, and we go on runs, or we talk at length in certain contexts, having tacos or whatever, about what you're building.

One of the reasons I think that you and I tend to get along well and always have a lot to talk about is that both you and I are building organizations that don't fit within many conventional rubrics. There are these kinds of organizations that are only enabled by the dynamics of the Internet, the emergent dynamics of the Internet, and both of us are working on platforms that essentially create and disseminate information about the Internet. I think those two things are closely intertwined, because we are excited about the Internet and its potential, and that's what fuels a lot of the passion behind both of our projects.

So freeCodeCamp, for people who are unaware is gigantic. It helps tons of people. As you said, we did a show four years ago where we talked about the basic rubric of freeCodeCamp, but we’ll obviously get into the architecture of it and the vision for in a lot more detail in this show. We’re going to do a long-form discursive conversation.

To lay the groundwork for that, one thing I want to understand is, is freeCodeCamp a business or a philanthropic endeavor?

[00:06:16] QL: FreeCodeCamp is a 501(c)(3) donor-supported, tax-exempt nonprofit. So it's public charity. The same designation that the Red Cross, the Doctors Without Borders, that a lot of these other big, multinational, nongovernment organizations have. So I would qualify it as a charity.

[00:06:38] JM: There are charities that are wildly profitable. There are very few of them, but Mozilla, for example. Would you rather freeCodeCamp be a donation-based platform, à la Wikipedia, or something more like Mozilla, where it's a nonprofit, but happens to be profitable?

[00:07:01] QL: Mozilla is a really interesting case, and from what I understand, it's almost a nonprofit that swallowed a much, much larger for-profit enterprise, like Firefox. I think Yahoo was giving Mozilla hundreds of millions of dollars a year to use Yahoo search as their default browser instead of Google, for example.

Mozilla is interesting because they’re so large because they’re almost a hedge against the more monopolistic tendencies of Google. So I don't know that Mozilla is a model that can be easily replicated by other nonprofits.

[00:07:39] JM: When you decided to build freeCodeCamp, you’re doing something without much precedent, but you were very confident that it would work. Where did you derive that confidence from?

[00:07:55] QL: Well, woe I wasn't super confident initially when I launched freeCodeCamp, but after people stuck around – So the brief history of freeCodeCamp, and I can go much farther into my personal history later, which I think is relevant. I have never really talked about it before. But this is the fifth anniversary of freeCodeCamp. I think that it's as good a time as any for me to explore that.

But just to answer your question real quick. After people started sticking around freeCodeCamp and after they started progressing through their curriculum and then going out and getting developer jobs, then in my mind this became a proven model, because people were able to, without spending any money, without even quitting their jobs or doing anything dramatically different than what they were already doing, just spending an hour a day coding on their own, they were able to transition careers. Once we saw a whole lot of people doing that, then it became clear that this was a repeatable model and that we could scale it.

[00:08:51] JM: Describe how your perspective shifted as you started to gain some significant traction with freeCodeCamp.

[00:09:02] QL: Well, we immediately knew that we could help a lot more people than we were if we continued to scale things up. We could've kept it small, and there's lots less risk in just keeping it small, and it's the equivalent of like hand assembling things by Ph.Ds in a clean room type environment, or we could go way out there and just try to help people who traditionally were not able to easily transition to the tech.

So initially, most of the people were here in America. Most of them were already college-educated, middle-class people, and gradually we've just tried to make freeCodeCamp more and more accessible to people outside of the U.S. and people who didn't necessarily have strong traditional foundational education that we would consider in the U.S. like going to a good public university, for example.

A lot of countries don't have public universities on the order of what we have in the U.S. A lot of people don't have access to even a secondary education that’s on par with what we have in the U.S. So we have people who’ve now gone through freeCodeCamp who grew up in the slums in Delhi or grew up in the slums of São Paulo and didn't have much education at all, but were able to use freeCodeCamp and learn enough that they could go out and get a job as a developer.

So that was a big jump that we took, and there was no real precedent to [inaudible 00:10:37] work, and even when it started working, you have to ask, “Are these isolated instances of extremely motivated individuals who just haven't had enough opportunity, or is this something that really can be generalizable to the point where freeCodeCamp can become kind of a curriculum of last resort for people regardless of their social economic background where they can potentially get a really lucrative job and provide for their families, and by extension, inject productivity and economic activity into their local communities?

[00:11:11] JM: You define yourself as a teacher. At its base, the idea of it teacher throughout history is a very esteemed role. It's a role that in order to be effective requires leadership, and leadership is complex, and difficult, and testing. There's a sense that the esteemed elements of the teacher in the United States for the last few generations has decayed a bit.

I think it has something to do with the increasingly mechanized process, the increasingly corporatized process of the public school system. I'm sure you have some thoughts on this. But what's at the core of your desire, the aspirant desire for Quincy Larson to identify as a teacher?

[00:12:25] QL: Well, teaching has a long tradition. Confucius was a teacher, right? You can go all the way back to ancient civilizations, and you can see teachers playing a role – Socrates. This is a tradition that is carried on through the generations, and the role of teacher has become more and more refined in modern times, and usually teachers are fitting into some sort of big school system. Whether that's at the university level or whether that's at a primary school. It's a pretty diffused role, but I would describe Carl Sagan as a teacher. I would describe Bob Ross as a teacher.

The actual definition of teaching is somewhat malleable. So I continue to use the title teacher, which I worked as a teacher for about a decade before I started freeCodeCamp. I continued to use that title, because think that teaching people through instructional design at scale is also a form of teaching.

[00:13:32] JM: Tell me about the decade of time you spent as a teacher.

[00:13:36] QL: Well, I will probably have to go even farther back to really tell you about that now I got into teaching, because it's pretty complicated.

[00:13:47] JM: We've got nothing, but time.

[00:13:48] QL: All right. Let's go way back. So I'm going to go way back to my childhood, if that's okay, and I'll bring you up to speed, because teaching was just one progression, one step toward ultimately what led me to create freeCodeCamp.

So my childhood ended when I was – It was April 19th, 1995. I was in Oklahoma City, 9:02 AM. I was in my science class and suddenly all the beakers started rattling and all the tables started rattling, and I'd never been in an earthquake before, but we just assumed it was an earthquake. All the students, teacher, everybody looked around and look at one another. Saw the room shaking and we didn't know what happened. A few minutes later we found out what had happened. There was building downtown Oklahoma City and a terrorist had parked a truck in front of it filled with fertilizer and he detonated it and killed about 160 people, including 19 little babies that were in a nursery on the ground floor.

So that for me was a huge jolt, and I went from just being like what I would describe as a pretty typical middle-class kid in middle America to starting the question reality and how we got to this point where something like this wasn't stopped. Then watching the ensuing chaos, watching how people around me reacted. I had a classmate, she lost her dad. Just that experience really woke me up. From then on, I couldn't really look at reality the same way.

So my dad was a strict disciplinarian. I got sick of his fighting, and my grandma had just died and I had inherited her 1986 white Ford Taurus sedan, and I grabbed all my stuff and threw it in the car and I just drove off, and I moved out and I spent the next year of my life just living in my car. Staying in the Walmart parking lots overnight where there was light, and spending – At first, I kept going to high school. But one day I was in the cafeteria and a man from my church who had been a huge inspiration to me. He was like always volunteered to lead Sunday school, and I was really good friends with this kid.

He had a college degree and he'd worked for decades. I saw him in the cafeteria with a hairnet on serving food, and he basically been reduced to doing that to be able to provide for his family. At that moment, I just – within that instant, I just kind of became disillusioned with school and I just decided to drop out.

So I went, I go to my car, and from then on I just started going to the library every day where it was warm and just rotating through all the different libraries around Oklahoma City and reading books and just thinking a lot and pulling up in the Walmart parking lot and going to sleep, and eventually I needed money to buy food and stuff. So I got a job at Taco Bell. My manager was this burly guy named Duke who had a very prominent Confederate flag tattoo on his arm that his uniform didn't cover, and I would just grab – Whenever business was slow, I just run to the backroom, I'd grab a burrito, build a burrito real quick and just eat it and just kind of contemplate my life and where things were going and what I was going to do. I really had no idea, but I decided that I probably did need to probably go to school if I didn't want to end up like Duke here.

So I took the GED and I enrolled in the cheapest state university I could, and I traded my Taco Bell job for working at different newspapers around town while I was studying liberal arts. I would just run around with my steno pad and interview people in different positions of power.

There were a bunch of rich homeschoolers around the city and I would just go over and teach them. So that was my first experience with teaching.

[00:17:53] JM: Wait. Sorry. Why are you interviewing people?

[00:17:55] QL: As a journalist. Basically, I work for a bunch of different local newspapers. So I'd interview hospital administrators. I'd interview people with the city government. I get to talk to all kinds of people, and that was really what gave me a glimpse into how organizations worked and power structures.

[00:18:11] JM: And laid the groundwork for your future work as a podcaster.

[00:18:15] QL: Exactly, yeah. That interviewing experience was really important, and the more I talk to people –

[00:18:22] JM: It's a great skill, right? Like it's super flexible.

[00:18:25] QL: Yeah. You can use it to – If you have a steno pad or a press pass, you can go in and talk to anybody and ask them questions and really get to the core of why they're doing what they're doing and how these different structures work. That was really instrumental for me. I really came into my own interviewing people and learning more about them and learning how these systems worked.

[00:18:48] JM: Not to take you out of your flow a little bit, but what was your mindset like in the early formative days of this adventure? Because I think when I – Not to take the spotlight off you, but like when I think about my kind of – I went through a lot of like random odd jobs. I've started so many random projects that have gone nowhere, and I think our audience, probably, there're a lot of people who are kind of going through this semi-random walk right now where they feel like maybe they're in a position where they know what they don't want. But they're in the transition from knowing what they don't want to building positive habits and iterating towards what they do want.

I find that that can be psychologically difficult, but also a psychologically bolstering experience. So I'm kind of wondering where your head was at as you were making your way through the library and Taco Bell and getting your GED at a cheap college, etc.

[00:20:04] QL: Well, for me, things were just so chaotic. The way I view the world was there's disappearance of order, but if we drill a little deeper, maybe can see the chaos underneath it. Why do these structures exist? Increasingly, I started to see like why our organization structured in a hierarchy where there's almost always one person at the top and then several layers beneath them, and people show up in the office at a certain time and they do a very specific task and they have deadlines and everything is measured in a certain way. Everybody's compensation is usually within one order of magnitude of the others, unless you get to these really large corporations where you have executives way at the top that make several orders of magnitudes more than the other people.

But I started to think about why. There was a reason why these kind of emergent properties had taken hold. So it was enlightening. I mean, it was really – I felt like I was uncovering some important stuff, especially when I was grilling these different administrators and asking them a lot of questions, learning about how they think. Why things were the way they were? The more you ask questions, the more questions reveal themselves. It was just this unending onion that I was peeling away layer to layer.

[00:21:32] JM: It's gets bigger as you peel it.

[00:21:34] QL: Yeah. I don't think that there is like some big conspiracy or anything. I think it's just a whole bunch of people that are operating in this system that miraculously works as well as it does. But it's probably not anywhere near as effective as it could be. That was the thing – I guess if I had to describe my experience, it was absolutely invigorating. It was just this euphoria of discovery, like, "That's how that works. That's why things are the way they were." But at the same time, it was very disappointing that 200,000 years in human history and we've got a bunch of people sitting in suits at desks sipping coffee and shifting papers around.

[00:22:18] JM: Well, to take this further, contrast the invigoration of your active learning process to the stagnation of the traditional education system that you grew so disillusioned with.

[00:22:34] QL: Well, I can totally see why education is the way it is. I mean, through years of just teaching and interacting with students, interacting with teachers, interacting with administrators, I started to understand that things were the way they were because they were good enough and I kind of came to see a lot of education as childcare essentially. It's the government's way of essentially relieving parents of their kids for the day so the parents can go and work.

Kids do learn stuff, but it's very suboptimal. I mean, summer break, you basically wipe out months of learning over the course of the summer break. Why do we have still have summer break? Having a classes a day where they're each 50 minutes long and kids spend almost so much time transitioning from one class to the next as they do actually sitting down and learning. There are all these things that reveal themselves.

The incredible cost associated with both private and public education. The average student in America, I think we spend like $10,000 to $12,000 per student per year for education, and this isn't just like government inefficiency or anything like that, because private schools are much the same. The cost are pretty similar, it's just that they've shifted to the parents and away from taxpayers where people are paying indirectly.

So that, and then when you get to the university level, costs are even higher. For some reason, we now have one administrator for every professor. Just the administrative creep and all these things, like how do you explain the growing bureaucracy? I think a lot of it comes back to human nature. I don't have any particularly satisfying answers for you. I can just say that it works the way it works because nobody has forced through a step change yet.

But I think that such changes are probably inevitable, and I don't delude myself to thinking that I personally can have an impact on this system that is everywhere. Every country in the world pretty much has a similar education system and has a similar higher education system.

[00:24:45] JM: Well, you are painting the contours of freeCodeCamp in a whole lot more resolute light through telling your history here. I'm starting to understand a little bit more about where you're coming from here. At this point, you're in this transitory state where you're doing interviews as a journalist. You're going to school. Let's continue down memory lane here.

[00:25:23] QL: Yeah. So I graduated, and by the time I graduated, I already knew where the big change was happening. It was the early 2000's and there was only one destination that an ambitious person would seek out, and that was China. China was undergoing a historic state of hyper growth where 300 million+ people were going from subsistence farming to middle-class service sector type jobs. I wanted to understand that.

So I found an international graduate program, essentially, where I could go and study with a bunch of Chinese. I got on a plane and I flew to Tianjin. I didn't know anybody there. I was the only Westerner on a campus of 20,000 people. I had a bunch of Korean international trade students who shared a room with me, and I set to work learning Chinese and learning as much as I could about China. I met my wife there. She was one of my classmates in one of my classes in grad school, and I spent the next two years of my life just learning about China and learning about business through the eyes of China. Learning about the history, learning about the people, and of course spending a huge amount of time just learning both Mandarin and Cantonese, because my wife is Cantonese. We spent a lot of time in the South. So that was the next period of my life, and it was an exciting time to be alive.

I was absolutely in the right place at the right moment to learn an incredible deal, and that will probably remain the most invigorating and exhilarating part of – I would say, it’s probably kind of capping off my childhood really was when I came back from China and I started working as a teacher and a school director here in the U.S. after my wife had been granted a visa. Eventually, she was able to become a U.S. citizen. Then I was running schools here in the U.S.

[00:27:26] JM: You were how old?

[00:27:28] QL: I was 26. I was the youngest school director in the school system, and I was just learning about how to run a school now and how to –

[00:27:38] JM: How do you get there? How do you get that job? How do you get the job as the youngest school director in a school system?

[00:27:43] QL: Will, it wasn't – When I immediately came back from China, I worked at a convenience store and I was trying to find any job I could that was focused on an international education. I found this one school that was an intensive English program on a university campus, and that's – I just did everything within my power to get that job. I learned everything I could about the people who worked there. I learned everything I could about the history of the organization, and I just went in there like a walking, talking encyclopedia, and against all odds, I managed to get that job.

I remember I interviewed for like six hours on three different occasions, because the director was – So she was skeptical that I would be able to do it, because I didn't have any real managerial experience. The teaching experience I had was mostly teaching homeschoolers and teaching engineers in Brazil, or from Brazil, in China. I would teach them how to use English, because there were just huge swaths of people moving to China to be able to participate in the big economic boom that China was experiencing.

So I was teaching all these ex-pats how to speak English and I was teaching Chinese how to speaking English, and so my teaching experience was fairly limited. I didn't have like a master's degree in education or in teaching English as a second language or anything like that. So I really just had to counter – I had to offset all that by just interviewing extremely well and being extremely knowledgeable. That was a big step, because before that, that was my first step actually into a managerial position.

If you think about what I do today with freeCodeCamp, it's mostly managing teams of volunteers, and then we have a small team of seven people, including myself, who work full-time on freeCodeCamp. So this is where I learned how to manage people. I was not only managing teachers I was also managing other administrators. I was managing homes. They're called like home stays.

Basically, people who open up their homes to welcome students from abroad. We had a lot of students from China, Japan, from Korea, from Saudi Arabia, from Brazil, and from Europe, and I'd have to work directly with those parents and iron out a lot of the problems. I'd have to work with agencies who wanted to place students in the U.S. I had to work with the U.S. government and the immigration department and just do an incredible amount of paperwork and basically keep this school running. So I think I've managed like 20 – I went from managing zero people to managing 25 people overnight when I get this job.

[00:30:25] JM: This is this is also around the time when you started to get serious about technology, right?

[00:30:30] QL: So after a few years of running schools in the U.S. and China, I was in a situation where I noticed that my teachers were spending a whole lot of time chained to their desk doing menial back-office tasks, like grade reports, like immigration reports, like attendance reports, things that could be automated if I learned some basic things. So I just Googled around. I learned some Visual Basic. I learned some different tools for like programmatically clicking through government visa forms and things like that, and I essentially told my stuff, "Hey, I'll take it from here," and they were able to spend all their time with the students, and the students were so happy, because that's what they came over here for, to learn English and to spend a lot of time with Americans and get ready to integrate so they could go to their graduate programs.

So that was the time that I learned very basic programming and scripting. From there, I saw how powerful it was and I started thinking, "Wow! I could build an entire system that helps schools run more efficiently." Then a thought occurred to me, "Well, maybe I should just learned how to program properly and then I'll be in even better position to build this tool." So I spent nine months just going to hackatons, hanging out at Hackerspace and coding and doing everything I could working through tons of textbooks, working through free online courses that were coming out of MIT and Stanford, and just every weekend, different hackathon. I did dozens of hackathons that year, and then I was able to get an entry-level developer job.

When I worked as a developer, that's when the real learning started, and I started learning more and more about working in teams, about using version control, about software engineering principles, about the dynamics of creating software and maintaining software during deployments. Fighting with weird bugs and things like that, and that was the foundry where my developer [inaudible 00:32:33] were forged.

After that, I still wanted to help other people. I wanted to build the school management tool, but then I realized, “Well, if I can just help other people do what I did and learn to code, then they can go out and they can build school management tools, and they can build a whole lot of other tools that progress things and move things forward, create new industries.”

So that's when I decided, “Well, I’m going to learn how to cope, or I’m going to teach other people how to code.” So I started building different projects around technology education, and eventually after several failed projects, I built the MVP for freeCodeCamp over the course of a long weekend. I pushed it live. I started tweeting about it, and people started showing up and using it. That was a really amazing experience, because that was the first I had ever actually built something from nothing that people cared about after years and years of building projects that I just couldn't get people to care about.

[00:33:29] JM: So I think that a big part of what makes freeCodeCamp successful and what will make it continue to grow is there is an aspirational vision to it, and I don't quite know how big that vision is. My sense is that it is big and grows every year, but it's possible that even when you are getting this thing off the ground, you had a vision for what freeCodeCamp could be that is more titanic than what has come to fruition today.

So before we start like diving into freeCodeCamp in gratuitous detail, I'd really like you to pull on the different threads of your unique experience and describe how those came together into the vision for what you are trying to do with freeCodeCamp. I really want to understand the vision and where you are trying to get to with freeCodeCamp.

[00:34:46] QL: That’s interesting question. So I would say some of the threads that come from my experience, first of all, questioning why things are the way they are. That was something that I learned pretty early on, and that informed freeCodeCamp, because I was able to ask a lot of questions like why do people meet in-person for school? Why do people have this set timeline? Why is school structured to where there is a beginning and an end? Why is there so much reading and some much lecturing in education versus just sitting down and doing it?

Then I started to realize, “Well, a lot of the reasons for these things are rooted in convention.” A lot of them are motivational, because if you're in a building with a whole bunch of other people sitting next to you, there is extrinsic pressure to perform. If you're just on your laptop after a long day's work, there's not the same pressure for you to actually do the work. So that informed a lot of freeCodeCamp, just understanding the limits of interactive, online, self-paced learning and understanding the motivational challenges that go along with it.

[00:36:00] JM: The MVP of freeCodeCamp, it was a certain way of learning to code, and I believe it has advanced quite significantly since then. I'm trying to remember what the first version was. Can you describe how freeCodeCamp has advanced since its MVP that you built?

[00:36:27] QL: Sure. For every listen when I say MVP minimum viable product essentially the first version of freeCodeCamp. So, initially, what I built over the course of that weekend was a curriculum where you could progress through. You could sign in and you could mark things done, essentially. Once you marked all of the things done, then you were done and you got this certification for freeCodeCamp.

So the curriculum was mostly external resources, like go take Stanford's computer science 101 class. Here's a link to it, and just work through the classes and then come back and check that you've done this once you finish it. We did the same thing with a lot of other different free online learning resource that were interactive. That was the first version. Important to that first version was I also set up a HipChat room. HipChat is chat room tool, and we would talk in there.

When people joined that chat room, I would be hanging out there and my goal was to motivate people and keep them progressing through there. So I talked to them. I'd say, "Hey, tell me a little about yourself. What inspired you to learn to code?" Leading questions that would waken within them that critical thing, like, "Hey, I should sit down and do this. I could transition into tech."

So a lot of the early days when we're just trying to inspire people, and two great things happened. First, people would also see what I was doing and they do the same thing, and they'd be right there alongside me welcoming people to the community and asking them questions and helping them when they got stuck or had any kind questions, or had any moments of crisis of confidence that they could reassure them.

So that chat room was absolutely instrumental, and people helping me with this process was instrumental. Then people started getting jobs, and that was the tipping point. Once people started getting jobs, this became a proven thing. This was no longer a toy curriculum that people did merely for self-improvement. It was an actual tool that they could use for economic development for their own personal career gain.

Once they started doing that, people would get a job and talk about how it helped them get a job, and people started contributing to freeCodeCamp. We slowly replaced the curriculum of external resources with our own homegrown open source BSD-licensed curriculum. BSD 3 three is one of the most permissive open source license alongside the MIT license. We used Creative Commons share like the BYSA. It's one of the most permissive Creative Commons license on all the curriculum.

So we were all working on these things together and building up this giant curriculum that, in my opinion, was initially inferior to a lot of the external resources that were taught by university professors, but quickly came to be of comparable quality. The freeCodeCamp curriculum had one huge advantage, and it wasn't some professor talking at you through a video. The entire time, you were in a code editor coding and you had this test suite, and whenever you thought your code was ready, you just hit control and enter or click the button and it ran all the tests and told you whether your code was indeed ready.

You just got into this flow state where you're progressing from exercise to exercise and learning experientially as you're building up projects. So that was a huge breakthrough in terms of motivation, was we sensed that a lot of the problems with traditional learning were just a really loose feedback loop. You sit in a class all semester and take notes and study those notes and then your evaluation criteria comes at the very end when you take the final, and you find out what score you got. It's hard to think of more course feedback than that.

But that's what traditional education generally has to offer. Yeah, maybe there are some quizzes along the way. But interactive exercises were – You're not waiting weeks or months for feedback. You're waiting a few milliseconds, because this entire platform runs right in your browser. It's not even sending data back to a server. It's just running the code right in your browser against a test, and within a few, often, microseconds, you're getting a feedback.

There’re no millisecond delay, it even hits the server. So it's about as tight a feedback loop as you can get, and that just really got sticky and people started implementing more and more challenges and the curriculum grew and grew from there.

[SPONSOR MESSAGE]

[00:41:23] JM: Today’s show is brought to you by Heroku. I have used Herroku personally more than any other cloud provider. I trust Heroku and I know that I can be fast and productive with it. As a developer, you're always looking for better and quicker ways to iterate on your code and add value for your customers. With over 9 million apps created on Heroko, over 2 million managed data services and with query requests served over 26 million times per day, Heroku has earned the trust of developers, and it's easy to start today as it has always been. You can try Heroku for free today by visiting softwareengineeringdaily.com/heroku to get started.

Our very own site, softwaredaily.com, runs on Heroku. We've trusted Heroku for years to run this dedicated website, and it's always been easy for us to be productive with Heroku. If you want to check out Heroku for yourself, visit softwareengineeringdaily.com/heroku to see it for yourself and get started.

Thanks to Heroku for being a sponsor.

[INTERVIEW CONTINUED]

[00:42:37] JM: So I remember this time when you and I met for the first time in person. I think it was at a Berkeley taco place, or I think you had a burrito, because apparently your Taco Bell days locked you into burrito mode. I think I remember actually a core answer where you also mentioned that much of your daily routine is built around a microwave burrito routine.

So burritos run deep in Quincy Larson's life. But I remember talking to you at this Berkeley burrito place, and I think I was a year – No. I might've been 9 months or 10 months into Software Engineering Daily and you were I guess probably a year and a half 2 into freeCodeCamp. So we were at kind of like similar junctures where we could feel a sense of traction, and I think you were listening to my show a fair bit around that time, because back then there were just really many software engineering podcasts and people in the states of voluminous isolation, such as yourself and myself, listen to a lot of podcasts, and you are really going down the rabbit hole of understanding software. So I think you are power listener around that time.

So we were having conversations about our respective project/businesses/Internet byproducts, Software Engineering Daily and freeCodeCamp respectively. I feel like both of us at the time were really unsure of what exactly it was we're trying to get to. I mean, even back then I think I was like, "Yeah, I mean I kind of hope all sell to O'Reilly, or like maybe start like five other podcasts and be like a twit kind of thing." I guess those are still like possibilities or something.

But, I don't know. It just felt early relative to where I think we both are at today where we have a little more confidence, a little more vision, a little more faith that the momentum or the inertia of our platforms will continue to sustain. I mean, I'm still paranoid. You strike me as somebody who's still paranoid about things just falling down.

But I guess I'd like to go back to that time. Because I remember you talking kind of like, "Yeah." I mean, kind of worried about Treehouse or – What were the other ones around that time? Code Academy, Pluralsight, kind of these other education competitors. I think the market has borne out that online education about coding is just bigger than anybody expected. It's big enough to sustain all these different verticals. Each of them have different competitive differentiators.

But I guess I would like to go back to that time and understand how your vision for the defensibility of the business, or not business, philanthropic endeavor, the finances. Take me back to the early days and give me a bit more refined understanding for how the structure of freeCodeCampus is a sustainable, defensible entity was maturing in the early days?

[00:46:06] QL: So early on, I was confident that this would be something that anybody could just come and replicate. Technically, it is. I mean, you can still build freeCodeCamp today. But one thing that we had that was really propelling us was this huge community of people who had either graduated from freeCodeCamp who are now working in developer roles or contributors who maybe had done that, but that were still contributing. So wants the community itself.

So freeCodeCamp as a software project, nothing terribly special about that. I think it's very well-built. I think the pedagogy that we baked into it is very astute and it works. It works remarkably well considering that you spend zero money and that you can do it at your convenience over the course of years. We have people who spend three or four years. I regularly see tweets from people said they started in 2016 and they just finished the final certification in 2019.

So the actual curriculum and the actual platform, nothing that exciting about it. But the community, that is what is really exciting, and it's difficult to build a community like freeCodeCamp has built up over the years, because it takes an incredible amount of hard work, and freeCodeCamp is not some corporate subsidy. It is not something that’s just bankrolled by a billionaire. It is a community of like-minded people who have similar career goals and similar passions who came together and started building together and built this tool up over the course of the last five years.

So that community itself is by far the most important aspect of freeCodeCamp and the thing that I think other people would have trouble replicating. I say that because it took an incredible amount of work. I spent the past five years helping people answering hundreds of emails a day, going to different study groups around the world and talking to people and trying to cultivate as much leadership as possible as you can within a primarily volunteer-driven organization.

So I would say that I'm not the best developer in the world. I’m probably not the best on the spot like go up and give a lecture style teacher in the world. What I think I am really good at and what I’ve worked really hard to become good at is leading a community and understanding the needs of the people in that community, understanding how we can help those people and how we can help those people help other people.

[00:48:48] JM: What's been the most difficult part of building freeCodeCamp so far?

[00:48:54] QL: The most difficult part is absolutely just keeping things running in terms of money. I am profoundly, profoundly bad at asking for donations, at asking for help, and it’s really shown, because a couple of weeks ago we had an outage. One of our API servers went down for two days and we did a root cause analysis, as we always do whenever there's some sort of outage. We did this postmortem. At the end, the only conclusion we could draw, the real finger was pointed back to me and the fact that we are running this organization on a shoestring.

FreeCodeCamp is currently I think in the top 1,600 websites on earth. We're bigger than TechCrunch. All these online learning platforms you mentioned earlier, Udacity, Treehouse, Code Academy, we're significantly bigger than all those in terms of usage, people using freeCodeCamp. So how are we able to do all this with a budget of $373,000 in 2019? How are we able to deliver 1.1 billion minutes of instruction that's equal to 2,000 years of people learning this year? We're not able to do it as well as we should be able to, because we had to cut a lot of corners. Everything has to be done as inexpensively as possible.

So we concluded at the end of that root cause analysis that we probably could have just spent 30 bucks a month on monitoring software that could have detected the specific type of outage and we could have just fixed it and it would've been two days of downtime. There wouldn't have been this big investigation. But when you're broke, a lot of times the check engine light comes on and you don't have the resources to pull over and have some mechanic look at it, right? That's precisely how freeCodeCamp has been operating, and that needs to change.

So that has been absolutely my biggest flaw as a leader, and the biggest problem with freeCodeCamp is that I am terrible at fundraising and I need to get better at it. I would say that, sure, I could just hire some corporate fundraising person to go out and get a bunch of donations and foundations to support us. But we are a grassroots organization. We have more than 5,000 people in the community who every month donate to freeCodeCamp. They donate 5 books a month or 10 bucks a month or 35 bucks a month. We want more of that.

We have more than 40,000 people who've gone through freeCodeCamp that are now working at tech companies. A lot of them are at Apple, or Google, or Amazon, or Microsoft. Some of these big companies with big salaries. I need to figure out a way to remind those people gently like, "Hey, if you can support us, we can help a lot more people here."

[00:51:48] JM: You need an alumni association.

[00:51:50] QL: So we have a LinkedIn alumni Association with like 60,000+ people in it. I'm just really bad about admitting that we need money, and I'm bad about asking – I'm bashful about asking for money. That's one thing I'm working on. That's the biggest bottleneck right now.

[00:52:07] JM: These universities are falling apart at the seams. Maybe you can pick up some alumni booster who's recently laid off. Those alumni associations are always hawking the most ridiculous, like, "Oh! You got a bumper sticker for $150 donation." Like, "Wow! Okay. That's a great deal. I get to say I was a freeCodeCamp alum if I make $150 donation." You need somebody like that to give these absurd donation.

I don't know. Speaking more frankly. I mean, you have people like Swicks who has given $10,000 to freeCodeCamp, and he's fairly new to software development. He's still making his way into building a career for himself. But he already sees so much value in freeCodeCamp that he's willing to give back. So presumably there will be more people like this, especially as you get like people who make their way into entrepreneurship.

[00:53:15] QL: Yeah, I guess we only need like one Bill Gates or one Jeff Bezos to go through freeCodeCamp to potentially be able to support the organization. That said, I mean, Shawn Wang, Swicks, he gave a $10,000 gift to freeCodeCamp a couple of days ago and he tweeted about it last week. He tweeted with this #payitbackwards, and he wrote this nice article about his reasoning for donating to freeCodeCamp, and I encourage everybody to check that out and to just search for the #payitbackwards. The holiday gifts giving season is ramping up and I'm going to go into full fundraising mode and try to get as many small donations as possible from different alumni. But, that's absolutely something that I'm going to do.

I totally see what you mean about grabbing some person from a university, but it's a very different situation.

[00:54:11] JM: I was kind of joking. I mean, that wouldn't be good for your brand.

[00:54:13] QL: So freeCodeCamp so for, every single person we brought on the team, all seven people who work full-time for freeCodeCamp came up through freeCodeCamp. I've never had to do a job interview for freeCodeCamp, because I've already known exactly who in the community is contributing the most. Whom can we give some additional resources to? Whom can we pay not to work full-time on freeCodeCamp, but to be able to devote themselves full-time to contribute to freeCodeCamp. Those are the people we've hired. So if I were able to identify somebody within the freeCodeCamp community who can help me with that, yeah, maybe that'd work.

One issue is because we're running so lean, where we have maybe a of couple months for the payroll in the bank at any given time. It's hard to take a risk of bringing on somebody to do this role. So I'm trying to learn how to do all these stuff myself from reading a lot of books about it. I've been studying the meta-of fundraising and trying to understand how it all works. Reading all the books about nonprofits, and we're in kind of a unique situation, like whenever we try to get somebody to sponsor freeCodeCamp, they treat us like we're some sort of hackathon, or whenever we're trying to get –

[00:55:23] JM: You mean when you try to get a corporation to sponsor you?

[00:55:25] QL: Yeah. Whenever we're trying to get, for example, credits to use, like in-kind donation from a company. A lot of times they're like, "Oh, is this a hackathon? What is this? What is freeCodeCamp? What's something like freeCodeCamp?" I have to go on this long explanation of what freeCodeCamp is and why it is the way it is.

There's a lot about freeCodeCamp that sounds unbelievable. I consider that somebody who's inside freeCodeCamp, it seems weird that there is this organization that millions and millions of people use that tens of thousands of people have gotten jobs as a result of using so far, and that the entire thing runs off of the equivalent of like one Silicon Valley engineer's annual salary.

[00:56:10] JM: This is why I actually led off the show with the analogies that failed to do it justice. You're an aberration. You're not Wikipedia. You're not Mozilla. You're not Craigslist. You are your own beautiful aberration.

[00:56:29] QL: So a lot of this was just communication challenges. How do we communicate what freeCodeCamp is? How do we communicate the scale? So one thing I did this year when I was preparing for our donation raising run was try to figure out how I can blow freeCodeCamp's sheer capital efficiency, if you will, down to a concrete number.

So I took that 1.1 billion minutes and I divided it by our $373,000 budget, and I came up with 50 hours of instruction provided for every $1 donated. So this is convenient way, because when people give us $5 dollars a month we can save, "You're paying for 250 hours of instruction every time you donate. That's enough for a classroom full of people essentially to be able learn to code for free just from your meager donation. If you give more money, it's equal to an entire village." So that's kind of my thinking in terms of how I'm planning to go into the holiday season encouraging people to donate, is just quantifying the direct benefit of their donations.

[00:57:43] JM: Okay. The $373,000 budget, that's the rough annual budget, right?

[00:57:50] QL: That is our exact budget for 2019.

[00:57:54] JM: For 2019, like thus far or projected to be the totality?

[00:57:59] QL: Almost all of our costs are fixed. So I can project that with pretty good certainty. That's how much it will cost to run freeCodeCamp for the remaining two months, plus the 10 months we've already gone through.

[00:58:12] JM: The big cost, are they developers or servers?

[00:58:14] QL: Both. But in any software organization, people are usually the main cost. We do spend several thousand dollars a month on servers.

[00:58:24] JM: I mean, that seems like – The servers at least, it seems like you could just get a company to – If that's relatively de minimis, then it might almost be like not worth it to deal with the overhang of like are you advertising Azure or like are you in bed with Amazon or something like that.

[00:58:46] QL: That's been our philosophy. We're happy to use – Amazon gives nonprofits a certain amount of money every month, or every year. We're able to get like I think a thousand dollars, maybe $2,000 credit from Amazon, and we use that in one month just on the emails. I send 2.5 million emails a week for our mailing list. So that very rapidly becomes thousand dollars a month just on sending emails. Using Amazon SES, which is the cheapest email service known to man. It's like you can send 10,000 emails for a dollar. So we quickly burn through all those credits.

Microsoft Azure gives us pretty generous credit as a nonprofit. We burn through that. Even with all those credits, we'll probably going to need to figure out some – We use a lot of other tools. For example, Algolia gives us search, and that means we don't have to run an Elasticsearch server. Algolia is in many ways faster than Elasticsearch and it's certainly easier to configure.

We've got all these in-kind sponsors who help us. I think they help most nonprofits, but it's a big help to us. So I don't want to diminish those in-kind sponsorships, but we don't want to be taking huge checks from large companies for precisely the reason you just said. We don't want to be freeCodeCamp brought to you by Google.

[01:00:06] JM: This is one of the things that like I think one reason I like talking to you is because whereas you are totally uncompromising in your kind of non-conflict of interest relative to corporate sponsors, I'm very much the opposite. I'm all about sponsored content. My show is riddled with ads. But I guess I I'm at least like transparent about it. Transparent to the degree that like I kind of can be without impacting the quality of the show. But like I feel that in my own behavior at certain times, can I really talk bad about company X when company X is a sponsor? I think it speaks to your long-term vision that you're uncompromising there.

[01:01:04] QL: What you're working with is a very different model. When you're a donor supported nonprofit – I mean, imagine that you're somebody who's giving to John Hopkins University and you're like a doctor or a lawyer and you're like really proud of your $10,000 donation that you just gave to John Hopkins. Then Michael Bloomberg swoops in and gives a $1 billion donation and its 4 or 5 orders of magnitude larger than your donation that to you was a big amount of money. I could just imagine it being really deflating, like saying like, "Oh! Well, should I even bother giving to John Hopkins anymore? I mean, look. They just got all that Michael Bloomberg money."

So freeCodeCamp, I'd like to avoid a situation where people feel like their donations don't make a big difference. What's the point of giving to freeCodeCamp if Bill and Melinda Gates are already giving them plenty of money? For me, personally, I want to give my money when I donate to causes that genuinely need it, where it's going to make a huge impact. I'm not directly practicing effective altruism. I know Haseeb and a lot of people whom you've had on the show are practicing – It's effective altruism.

[01:02:15] JM: Effective altruism. Yeah.

[01:02:16] QL: That's where you give some percentage of your salary to a nonprofit and you've made a commitment to you're going to distribute, and there are all these organizations that rank charities by their efficiency. FreeCodeCamp, we have a platinum transparency rating on True Star, I think. But there's like a great organization called Donors Choose, and that's what Haseeb and a lot of other effective altruism people use. That organization basically ranks charities for their efficacy and helps you maximize the amount of impact that each dollar you give is going to have.

Unfortunately, because freeCodeCamp is so new, we don't have the seven years of tax returns necessary to be able to qualify to be on those lists. So that's one thing that eventually will come and save us, I think is because I am comfortable that we'll be able to make it on those lists. I'm confident that, as you said you said, we're going to start having a lot more alumni who are in a position where they can donate a lot of money. FreeCodeCamp will eventually be able to have a lot of resources that we can use, like the Red Cross, and like Doctors Without Borders and a lot of these other nonprofits that are doing tons of work and are able to employ a lot of people toward the cause that they've gone after.

So I'm confident freeCodeCamp will get there. It's just the kind of rocky intermediary period where everything is held together with duct tape that I'm worried about. Now freeCodeCamp, we could in theory lay everybody off. I could lay myself off. I could just go get a job somewhere and pay for the servers, and I do think that me working somewhere, I could probably afford to keep paying the servers and keep freeCodeCamp on life support. Absolute worst case scenario. So I'm no longer worried that freeCodeCamp will just blink from existence. That's not going to happen. The real question is how much potential to do good in this world are we foregoing by being so undercapitalized. That's what keeps me up at night, the miss-potential.

[01:04:14] JM: We won't go much further on this. But if Bill and Melinda Gates were like, "You know what, Quincy? 2 million bucks a year. You got it." Would you accept?

[01:04:25] QL: I'm not sure. I can't definitively say like we would reject that out of hand because there's the possibility that a lot of our donors would be like, "Oh! You should give that." But if it meant that accepting that money was going to cause our small grassroots donations to go down dramatically, I'd probably turn it down, because in the long-run, it's a much more robust strategy for an organization like freeCodeCamp to have tens of thousands of stakeholders who are donating every year than to have one or two large foundations that control its destiny.

[01:04:55] JM: How has the market for coding education products changed since you started freeCodeCamp?

[01:05:03] QL: I think that more and more people have realized that you can get a good job if you learn to code. Things haven't changed enough though. I'm surprised at how little things have changed. If you look at the number of computer science majors in the United States, it hasn't gone up dramatically. It's gone up a little bit, but not in line with what you would expect.

I am thrilled that there are so many coding boot camps out there that are helping people in their communities learn to code. I'm surprised that there aren't more. I'm surprised that so many of them have gone out of business, because so many jobs out there are still unfilled that will require coding. Something like 60% of all STEM jobs require programing.

I have plenty of friends who are doing like PhDs in different fields, science, genomics, all these things, and they use programming almost exclusively. That's what they do. They take the datasets and they write scripts to parse those data reliably where they create pipelines to be able to parse data. Anybody who's working with data needs to know how to program, and yet things move so slowly.

That's one thing that’s totally taking me by surprise, is when I started freeCodeCamp, I was so sure that like the government was going to drop tons of money on programming education, or all the money that Google and all these other companies was dropping into education – Facebook, that this was going to make a huge difference and then suddenly there wouldn't be such a huge crunch for developers that they wouldn't be so scarce. That would be good. I think would be great if there could be a million more developers in America, or two or three more million.

[01:06:43] JM: There's got to be many more though. I think it's just the indexes aren't capturing it, right?

[01:06:48] QL: I'm not sure. I mean, there are definitely more developers. But you have to understand. I’m so caught up in the meta. I'm on Twitter seeing people posting that they just got a new job. This is kind of the tip of the iceberg. There are so many people out there that – And that's the crazy thing when you start thinking about macroeconomics and you start thinking about how there are billions of people around the world working and there millions of people just in the United States working cash registers and retail, things like that.

You start to think about all these people and then you think about the very, very, very tiny proportion of people who know how to code and how these people that know how to code could completely append these industries and make things dramatically more efficient and unlock so much economic potential, and yet it's not happening anywhere near as fast as we want it to.

[01:07:37] JM: What do you understand today about coding education that you didn't know at the beginning of freeCodeCamp?

[01:07:43] QL: I would say that it’s something that I dramatically underestimated was just the motivational aspect of it. Programming is not intrinsically a hard technical thing. I think there are probably a lot of subjects that are a lot harder to wrap your head around than how to create some basic algorithms and data structures, right?

[01:08:03] JM: That is for sure, man.

[01:08:05] QL: And yet, it is a very frustrating field to go into, because the entire time you're sitting there with some compiler or with some interpreter telling you you're wrong, you’re wrong, you’re wrong, you’re wrong. You’re getting error messages. You’re getting field tests. Things are not working. You're going and reading stack overflow discussions from 10 years ago and trying to get unstuck constantly. So it takes a lot of grit to learn to code and it takes a lot of grid to stick with it and not just to go out and farm goats or whatever a year or two into your career.

For me, programming is very much a motivational challenge more than anything, and that much has really been driven home as I’ve seen people scrub out of the field so to speak not because they couldn't do it, but because they didn't want to keep doing it. They were like, “Oh! Please give me – Oh! There’s a management job. I don’t have to code anymore. Let me jump to that,” right? So that's one of things we've been working very hard on, is how can we keep people inspired. We’ve got a section on the freeCodeCamp form called You Can Do This, and people when they feel the depth of their frustration, the depths of their despair, they can go there and people cheer them on.

I think all the meet up groups in different cities and all of the different study groups and all these organizations, like Women Who Code like Hack Club. They’re keeping people motivated and pushing forward through the despair and keeping them progressing with their skills so they can gradually acclimate to the strain of being a developer. There's no rule that says you have to eventually quit programming and you have to move into some managerial role, right? There are plenty of people out there who love programming and they've been doing it for decades. So how can we keep people motivated and keep people interested in technology and keep them from burning out?

[01:10:00] JM: I could not agree more with what you are saying here. The time when I first crossed paths with you on the Internet was when I was on Quora a lot. Part of the reason I was spending so much time on Quora was because I was really having a lot of trouble in my computer science education in college. I think one of the reasons I was having so much trouble is that conventional University computer science education is not built to inspire. It’s kind of built to reinforce this frustration and this ivory tower sensibility of, “Look, the compiler is telling you you're wrong, because you do not belong here, and you are an idiot.”

I felt this reinforced in the mannerisms of many of my professors, like, “Look, if you can't cut it, if you can't write the unit tests, if you can't get your projects done in two weeks, maybe you should drop out. Maybe this major isn’t for you. Maybe you should go to MIS. Maybe you should go back to business school. Maybe you should get out of here.” They really would take this attitude that like, “Maybe you should take a step down.”

That's not the attitude that an educator should take. The educator should basically take, “Look, what do we need to do to reshape your mind so that you can actually succeed in this field, because it's really not that hard.” Yeah, I mean there was something about Quora. There was something about just Internet culture. The Internet culture has the elements of inspiration that you need – I mean, Seth Godin, somebody who I’m a fan of. I found his podcast around that same time and he's all about basically like give the middle finger to these people who tell you that this is not something that's doable. It is going to take grit, but it's actually not going to take much more than that.

[01:12:06] QL: What you described with the university experience that you went through, that’s an unfortunate reality in a lot of universities today. Here in the U.S. and also oversees, there’s this kind of elitism. I'm not sure exactly why it is. It could be just rooted in the fact that university professors are fighting so hard for such scarce positions in academia.

[01:12:29] JM: And the math and physics lineage of computer science.

[01:12:33] QL: Yeah. But the reality on the ground, it couldn't be more different from that scarcity that those arbiters are trying to reinforce. The reality is there are incredible opportunities, and this is something not just with software engineering, but I think with medicine. Medicine has this almost kind of ritualistic hazing process they put all the physicians through, and medical school is brutal, and [inaudible 01:13:53] is brutal.

[01:13:54] JM: I don’t know if I’ve told you this, but I was premed until I took organic chemistry. There was this semester I took organic chemistry. I was like, “Okay. All right. Fine. I got vetted I'm out. Peace.” I would do the exact same thing. Yeah, the weeding processing. Sorry. I’m sorry. Not to take my seat at the couch.

[01:14:13] QL: It is really unfortunate that as of late 2018, so much of education is still like this. It's still some – Almost always a man who has gone, walk through the snow uphill both ways and wants to see to it that you do it too. What does this affect? Well, if you want to look at why the profession is so homogenous, it's because people who can't visualize themselves can't find people who look like them or come from backgrounds like them in the profession. They don't even have that glimmer of hope that you or I might have seeing some guy who looks like our dad, for example, who's in the field and saying, "Well, we can do that too." They don't even have that.

So there's even less to help empower through this process of weeding. So, yeah. I'm optimistic that if anything, freeCodeCamp can present competition to the University computer science programs and force them to become more inclusive and quit hazing people. FreeCodeCamp can essentially, by increasing the quality of what's available for free, force everybody else to up their game .If nothing else, I will be very proud of that contribution to education that we raise the floor.

[01:15:38] JM: Do you see a continuation from the education of computer science fundamentals, software engineering best practices, security data science, et cetera, these things? Do you see a continuation from those into perhaps lessons about entrepreneurship? Do you want freeCodeCamp to be a place where you can go full stack from learning to code to building a software company?

[01:16:16] QL: So here's my thinking on this, and I realize that this may be contrary to a lot of popular opinion and a lot of wishful thinking about entrepreneurial. I think that before you can be a successful entrepreneur, you should go out and you should work for other people and you should manage other people within an existing organization. Otherwise, you're going to be somewhat naïve and you're going to have to make a whole lot of mistakes on your own dollar or on your investor's dollar, and a lot of people are going to suffer from those mistakes who would be somewhat insulated from them if they were in one of these organizations.

So I spent about six years running schools, and that managerial experience was absolutely essential. freeCodeCamp would not exist if I were some fresh college grad who was trying to create a project like this. I think there's a reason why the median age of a successful entrepreneur is usually in their 40s or 50s, because they've been around. They've worked with an existing system. They understand not only how things are, but why things are the way they are. Thus, they are able to think within those constraints and make slight tweaks to those permutations and make their organization successful.

The key to success is not in bucking convention. It's in knowing which aspects of convention to keep in place in which aspects to discard, that selectivity. So the idea of freeCodeCamp becoming kind of like learn to code and become an entrepreneur I think is probably beyond the scope of freeCodeCamp, because I don't know the that's necessarily possible to do reliably. I do think that people should go out and get jobs. I think that is one thing the freelance, and that's a great way to get a job, but you really should work at a company with a team of software engineers alongside you and a product manager and stakeholders and customers and all those things so you can better understand the entire software development process.

I guess maybe if I'm a little more bearish on entrepreneurialism, it's just I'm cautious. I've seen people go out and destroy their life savings. I've seen people go out and destroy their reputations by taking too many risks to quickly. So life is long. Human lifespan ideally should be increasing. I'm almost 40 and I'm hoping to live a hundred, and I'm going to keep learning along this entire process, but I would encourage people who relatively early in their careers to keep working and learn on the side and keep learning as much as they can on somebody else's dime before they go out there and they try to do a risky experiment, because entrepreneurship is one – I think the definition of entrepreneurship is somebody who bears risk and you have to be ready to bear that risk. You have to have savings. You have to have a plan B if things go south. All those things will set you up for success more than just taking an online course and trying to throw yourself into entrepreneurialism, in my opinion.

[01:19:31] JM: I want talk about software architecture and software engineering, because you had this – As you said, this post about how what went wrong with freeCodeCamp servers. You also had this recent post about freeCodeCamp at five years. Your software architecture has gotten pretty complicated. You've got a lot going on. You've got serious engineering challenges because you are at scale. I mean, what are you? The 1700th most popular site according to Alexa rankings? Something like that?

[01:20:10] QL: I had to look it up, but we are currently 1589th as a recording.

[01:20:18] JM: That is indicative of a ton of traffic. The reason there's a ton of traffic is because you have a lot of content. You have a lot of curriculum. I'd like to talk – We don't talk in gratuitous detail about how the curriculum works today, because I think probably a lot of the people listening are students of freeCodeCamp, or people who have seen it before. But let's just talk through like the contours of freeCodeCamp at a high-level, like freeCodeCamp as a product, and the product surface area. Then we can dig into some of the interesting architectural problems that you deal with with a product that is at this scale, at this throughput, and with the small of a budget. So you have all these interesting cost tradeoffs to make as well. So let's first just talk about the product surface area.

[01:21:13] QL: Sure. So freeCodeCamp is really three different core applications. There's the curriculum and portfolios. That's one of the pillars, and that is all custom code for the most part. That's when we say the platform. That's generally what we're talking about. We also have freecodecamp.org/news, which is our publication, and we publish several articles a day there often in-depth explanatory journalism, biotechnology. Then we have the freeCodeCamp forum, and that's the third pillar of freeCodeCamp.

Now, all three of those run on separate servers. The freeCodeCamp forum is powered by Discourse, which is a really cool self-hosted open source forum tool created by the founder of Stack Overflow, Jeff Atwood and his team. That's very well-maintained, and you can get your own instance of discourse running really easily if you want to. You can just deploy it to the cloud and it will live on its own server in a Docker container, and there you go. Same thing with freecodecamp.org/news. That is running on Ghost, which is another self-hosted open source project that just released version 3 of their tool. It's like the WordPress, but it doesn't have a lot of the baggage that comes with WordPress. WordPress is excellent, but it does have some vulnerabilities and it does have a very complicated plug-in system, and WordPress is trying to do a lot of things, whereas Ghost is just trying to be a publication tool.

So we have maybe 7,000 articles on freecodecamp.org/news at this point, and we have hundreds of thousands of posts on freecodecamp.org/forum. So those are those two. Now, if you want, I can drill into what powers the freeCodeCamp platform, the core curriculum.

[01:23:17] JM: Please do.

[01:23:18] QL: Yeah. So we are running React on the frontend. We're running node as the backend. We're running a tool called Loopback, which is an API tool. It essentially turns your entire app into an API where all the routes can fetch JSON as well as HTML. It's something that's already baked into tools like Ruby on Rails. With Ruby on Rails, you could take any URL and you can just put .json at the end of it and get a JSON representation of the data and that route. So Loopback does a lot of stuff.

We use MongoDB for our database and we have a managed cluster that's MLab, and we're working on moving that over to MongoDB's new Atlas tool. MLab was acquired by MongoDB recently. So they're kind of sunsetting MLab, I think. So we're in the process of moving everything over. But it's good to have a big sharded system, because we have a lot of data. We've got millions of records, and each of those records has millions of challenge completions and a whole lot of other data that they've added to their portfolio.

The portfolios themselves, we're working on growing them and making them a lot richer in terms of just having a lot more data. You can submit a whole lot of different projects you've built from outside of the freeCodeCamp curriculum as well. If you built your own JavaScript game, for example, you could submit that. Turning that more into kind of like a LinkedIn light for developers. So we're working on that.

The platform itself in terms of the curriculum is a Gatsby app, and you hit that URL, freecodecamp.org/learn, and the entire curriculum loads and it stays in your browser, all of it, like thousands of challenges. Then when you complete challenges, it will post those to the server. If you disconnect from the server for some reason, you can keep completing challenges. Then when you reconnect, it will push all that to freeCodeCamp servers. So we're very close to having it be fully offline functional app, like a progressive web app in that regard. It's not quite there, but we may be just a few months away from that. We've been tinkering toward that for the past year.

[01:25:42] JM: That is sick. I mean, well the offline functionality, that something that developing countries – I mean, that's a necessary – To have a buttery experience, you need that kind of offline functionality.

[01:25:56] QL: What you just said about people in developing countries, freeCodeCamp is founded on a single principle, and that is access. That doesn't just mean accessibility in terms of blind people being of able to use freeCodeCamp. People with motor skill impairments being able to use freeCodeCamp. All those people can use freeCodeCamp. We've got a 100 Lighthouse score for accessibility, and we take that into consideration with everything we do.

But also people who have frequently interrupted power, frequently interrupted Internet access. So not only are we trying to build the PWA that addresses all those concerns and not only do we make our data as small as possible so the entire freeCodeCamp curriculum fits in a footprint of just a few megabytes. But we're also working on eventually getting an android app on, and maybe even an iOS app. Because so many people in India, so many people in Nigeria, other countries where a ton of people use freeCodeCamp, Pakistan, Kenya, places like that, we want to make sure that they are able to use it on their own devices, and people don't necessarily have laptops.

So we recently just completely overhauled the mobile freeCodeCamp experience, the mobile web version, to make it easier, put everything in a tab so you can move between your code, your tests and your HTML preview your console output. We're also working to make it to where it runs natively so that you can just download a freeCodeCamp app from the App Store and you're good. Then when you have Internet connectivity, it pushes all your stuff up and you can still claim the certifications and everything. But we're moving more and more of that into freeCodeCamp.

For example, right now, most of the projects are built on CodePen, which is awesome. We'd like to make it where all those can be built right on freeCodeCamp so you don't need to go to an external website in order to build your projects. We're also working with some code that Mozilla has just put out to be able to run Python in the browser too. So we're going to be teaching data science concepts with Python and we're going to be teaching TensorFlow, NumPy, Keras, a lot of these libraries in Python as part of our core curriculum.

[01:28:02] JM: I think the entirety of the backend architecture is too complex to portray to people over a podcast unfortunately, but it is an interesting backend, and I encourage people to check out your articles, your recent articles about it on freeCodeCamp News. But just to give him a little bit a context for the kinds of problems that you need to work through, can you tell me a particularly difficult engineering problem – I mean not the one that you outlined in your recent post, because you kind of laid that out in gratuitous detail, or maybe if there's details that you didn't layout. We could go further into that. But I just like to know a particular scalability bottleneck or some kind of difficult backend engineering challenge that is emblematic of the kinds of software engineering problems that freeCodeCamp encounters.

[01:29:00] QL: I would say we have of those backend-specific challenges from scale and from the way we're architectured and things like that. But perhaps more interesting would be the overlap between instructional design and engineering that we've had to navigate. So we very early on decided that we wanted everything to run client-side, like all the tests run client-side. That's to minimize the amount of latency. It's also to just minimize variability and things like that. So it's been a real challenge to get a lot of these subjects taught in a fully-interactive client-side tested way.

So some of the early things we had to struggle with, for example, is how are we going to teach, for example, Rest APIs and specifically how are we going to teach Ajax and things like that interactively in the browser. So those are things where we've almost had to tailor the instruction to the medium and the constraints of the medium. So that has been an ongoing challenge, and we've just had breakthroughs over the years as to how we could test different things in that. How we could –

So one of the big breakthroughs recently was – I think about a year and a half ago, we decided that the test suites, the projects originally, you didn't have test suites associated with them. You just have to build it and then we would manually review it and other people give you feedback on your projects. But there wasn't that tight feedback loop of like, "Oh, this project doesn't – This project does exactly what it's supposed to do and it looks the way it does."

So what we did was we created the this API endpoint you can hit and you can download a test suite right into your project wherever you're building it. Whether you're building it locally on your local computer, whether you're building it on a CodePen, whether you're building it on just like a GitHub pages app that you're deploying in Netlify or something like that. You can load this test suite and you can run it right against your code. So whenever you submit the URL, you're also certifying that it passes all of these tests. So there'll be maybe like 15 or 20 different tests in a test suite for these different projects.

FreeCodeCamp, if you all don't know, is essentially all projects. We have lessons right now. We're getting rid of those lessons and we're replacing them with practice projects where you learn JavaScript by building a role-playing game, for example. You build a fully-functional role-playing game where you're going like fight a dragon, get a sword, build up your health points and stuff like that. But you build that line by line with JavaScript. That's how you'll learn JavaScript, is building a role-playing game.

[01:31:39] JM: I mean, unless you want to drill deeper into that engineering problem, I kind of want to jump on what you just said, the idea of building a role-playing game rather than having these specific lessons.

[01:31:51] QL: Yeah. We always wanted to have fully project oriented learning, and to some extent, the existing freeCodeCamp curriculum, like when you build the first HTML app, you're building up kind of like a can photo app. But it's still a series of discrete lessons, right? The new curriculum that we're building, we're calling it version 7 of the freeCodeCamp curriculum, is totally projects. So instead of having a series of discrete lessons, instead you'll just have a giant code editor that will take up almost the entire screen and its Visual Studio code. We're Monaco, which is the browser version of Visual Studio code.

[01:32:29] JM: Oh, sick!

[01:32:30] QL: Yeah. So you get all the syntax highlighting. You get all the tab completion, all those functionality, right? You don't have the full IDE experience, but you can pull some of that in. Anyway, what it's going to be is there's just going to be a giant code editor and we're going to split the code editor open and there will be a test inside of that right above where you need to change the code or type a line of code. Them you pass that and, boom! The code editor closes up and then a new rift forms below on the next line, and there's a test. Everything is taught through a series of tests that you have to get past.

So you'll get hundreds of tests passing. Through the process, you'll build up this sophisticated application. There'll be a lot more repetition than there is currently. That's one of the big complaints about the freeCodeCamp curriculum, is that we hit a whole bunch of topics in rapid succession and we don't necessarily revisit them as much until you get to the certification projects. Well, we're going to try to engineer a lot of repetition in there to help – Even though recall is not nearly as important with software development as recognition is, and in certain fields, recall is everything. That's why the entire U.S education system is basically based around retention, right? You remember all these facts so you can pass this test. With software engineering, everything is only a Google search away.

But we're still – Because that's how people are trained in pretty much every education system on earth, they're trained that you have to retain things and you shouldn't – There's a discomfort associated with having a vague remembrance of something, but not having like concrete that you can recall. So we're almost – It's a concession to how people are or meeting people where they are, as supposed to where I our pure teacher's minds where they should go. But we are going to incorporate more repetition for that very reason to make people feel comfortable, because that's one of the most common reasons people give when they quit freeCodeCamp. It's like, "Oh, I didn't feel like I was learning anything. I didn't feel like I was retaining anything. I did all these stuff and at the end of the day I couldn't remember the exact syntax of how you make a Boolean declaration in JavaScript or something like that," right?

So there's going to be a lot more repetition, but you're going to build through a series of just discrete tests every single step, a single test that you have to get passing. Boom! You move on to the next one, and you're going to get into this flow state where every 30 or 40 seconds you're going to get test passing and you're moving forward, and hopefully like one sitting you're going to build out an application. So you'll build – Right now, the entire freeCodeCamp curriculum is 30 projects. By the time we're done with this version update, it's going to be closer to 100 projects that you'll build.

[SPONSOR MESSAGE]

[01:35:22] JM: Feature flagging makes it easy for your team to quickly change the way that your product works, and CloudBees Rollout lets you manage feature flags easily. When you have a solution to manage feature flags at scale, you're empowered to continuously and intelligently rollout changes as soon as they are code complete on any platform, even mobile. You can decouple development from code release for a real-time change control. You can rollback only the changes that you don't want, or keep them around.

You can toggle features. You can use multi-varied flags for A-B testing and you can remove misbehaving features with a kill switch. All of these is part of the feature flag platform that is CloudBeess Rollout.

Visit softwareengineeringdaily.com/cloudbees and try a free 14-day trial and experience how CloudBees Rollout can help you with every release. Visit softwareengineeringdaily.com/cloudbees to get a free trial. Cloudbees Rollout is trusted by large users, such as Zendesk and Jet.com. Try it out today at softwareengineeringdaily.com/cloudbees.

[INTERVIEW CONTINUED]

[01:36:46] JM: It's interesting that the move from having – Like I can imagine the initial lesson plan for freeCodeCamping, like, "Okay. First you do Hello World. Then you figure out how to parse JSON. Then you figure out a two string method or something." The kind of like boring boilerplate stuff. I think what I'm hearing is you're finding that the boring boilerplate material can be exhaust. That can be a side effect of starting with something that is exciting and fun to do, and perhaps adding a little bit of repetition, a little bit of a slower pace, but focusing completely on the project orientation, because that is more fun and that the motivation is so important there, and you will be able to get the boring boilerplate stuff just as an additive outcome from the project-centric focus. Am I understanding that correctly?

[01:38:06] QL: Exactly. My economics teacher when I was in grad school, he sat me down during one of our office hour session and he grabbed a book off his shelf and he started just –

He laid it on the desk and he started flipping through it and there was just a bunch of equations. He said, “This is my economics PhD. This hundred-page book has all the equations that we use for my economics PhD. I spent six years after undergraduate learning this stuff, and it all fits in a 100-page book.”

But that is just the raw facts, right? That completely disregards the human element of how we grapple with learning, how we retain, how we internalize those lessons, right? People don’t sit down and start memorizing equations. That’s not how humans work. That's how computers work. Computers are very good at retaining facts perfectly. Humans don’t work that way. Humans learn through experiences. So a lot of our journey with freeCodeCamp toward watching how millions of people interact, watching who decided to keep going, who makes it at the end. What powers them through? What kind of projects they build? How their projects are different from people who stop halfway through? All that has been kind of crystallized in this pure project-oriented orientation the we’re taking now with the big update to the freeCodeCamp curriculum coming in 2020.

[01:39:38] JM: Do you try to like test it? Do you throw it at new users in kind of a slow onboarding fashion, or do you have enough faith that this is the direction to move towards such that you don't feel like you need to kind of do an AB testing and gradual shift of the curriculum?

[01:39:59] QL: AB testing is really good for something, right? It’s really good for determining like what color of a button is going to get people to convert in your shopping cart, right? It’s less good for something where you’re trying to walk somebody through a very complex experience.

I have no doubt that when people say there's more of an art than a science to something. They don't yet know the sites of it. I have no doubt that there is probably some very good way to AB test and to arrive at a totally optimal solution, but I do doubt that we’ll be able to do that in a time effective manner. So to some extent, yeah, we’re just throwing this together, our best guess, and then we’re going to put it up and there we’re to see how people react with it. We have really granular data. We know exactly how many seconds on average people spend on a specific challenge within the freeCodeCamp curriculum. That's how we've identified a lot of the bottlenecks.

My philosophy has been people shouldn't spend more than two minutes on a given lesson and they should be able to figure out how to do it from the instructions and from the test messages, and they should be able to progress. So we find those outliers and we're able to go in and tweak the wording or maybe break it out into several lessons in order to make it easier for people to get through.

With these with these new projects, we'll be able to do the same thin. We'll be able to identify which tests out of the 200 or 300 tests associated with building this project is the stumbling block and smooth that out. As far as rolling out all the projects at once, a lot of it is not so much a teaching issue as it is a communicating issue. How do you communicate to the millions of people who are using the freeCodeCamp curriculum that, "Hey! This is changing. You're not going to have individualists lessons." Now you're going to build projects," and trying to do that in a piecemeal fashion where you're like communicating to one person, "Okay. These lessons went away, but now we've got this project." It's just a lot easier in our experience to rip the proverbial Band-Aid off than it is to slowly inch it off."

That's been our approach, is generally we observe. We figure out what we think the next step is and then we work really hard and then we ship that next step. It is to, some extent, in conflict with like Agile software methodologies and a lot of other findings that I think work really well for other types of products. But for our curriculum, it's better to proceed in bounds than baby steps.

[01:42:29] JM: The people who are writing the code for freeCodeCamp, what's the balance or the portion of time spent and difficult problem solved between – Because it's open source. What's the distribution between people who are paid employees versus open source philanthropic contributors?

[01:42:59] QL: FreeCodeCamp's team is seven people. I can go and list them real quick for your benefit. So that's me. I work full-time. Beau Carnes. He runs our YouTube channel. He does a lot of instructional design. He's been working as a teacher for the past five years.

[01:43:13] JM: By the way, instructional design, you mean developing and creating curriculums?

[01:43:16] QL: Yes. Instructional design is basically teaching at scale. That's how I define it. Baking the teaching into the materials. He's an experienced classroom teacher. He was teaching middle school for the past five years and he had worked as a software engineer for a while as well. He'd been contributing to freeCodeCamp for years, and then we were able to bring him on full-time.

Then Mrugesh Mohapatra. He's in Pune, India, and he is doing a lot of our major architectural decisions. He has worked at a bunch of different multinationals in India and has a traditional software engineering background, but he'd been contributing to freeCodeCamp for like three years before he came on.

Ahmed Abdolsaheb is based in Turkey. He is a designer and a developer, and he had also been working with the freeCodeCamp curriculum and contributing to freeCodeCamp for years before we brought him on. He does a lot of the design work. You will notice that freeCodeCamp has a fresh coat of paint when you go there. We have redone the design style guide and we've adapted a cell I'd to call Command Line Chic, which is heavily modeled after kind of almost retro competing aesthetics with contemporary accessibility sensibilities. So he does that.

So that almost rounds up our development team, but we also have Kris Koishigawa who's in South Korea, and he is a developer who work as an English teacher in Korea for the past six years. He contributing to the freeCodeCamp cocaine curriculum extensively, especially the interview practice preparation sections, writing test for like the Project Euler problems and Rosetta Code problems and things like that. He did that for a few years and now he's working full time with freeCodeCamp.

Kris and Beau are primarily focused now on curriculum development. Again, both of them have strong teaching backgrounds and both of them are able to contribute to the codebase, but that's primarily what they're doing. Then not working so much on the codebase, but working on the content side of things. Abbey, Abigail Rennemeyer. She's out in Portland and she is the primary editor for the freeCodeCamp publication, freecodecamp.org/news. She edits virtually every article that gets published there, thousands of articles a year. She also writes and she runs the podcast, and she also runs all of our social media.

Then Miya Liu is based in Hangzhou and in Chengdu in China, and she runs the freeCodeCamp Chinese community. The freeCodeCamp Chinese community is almost kind of like a parallel freeCodeCamp. We have hundreds and thousands of people who use freeCodeCamp over there, and we have tons of major communities and major cities where people get together and come together. So that's a big part of freeCodeCamp is what Miya is doing over in China. So that rounds up the entire team.

Now, in addition to them, we have dozens of monthly active code contributors who are contributing code directly to the freeCodeCamp platform. Some of them are also making incremental tweaks to the curriculum. Then we have a ton of contributors who are contributing to the publication. We have contributors who are contributing to freeCodeCamp's YouTube channel. freeCodeCamp's YouTube channel is now the biggest programming channel on YouTube, other than the New Boston, which kind of shut down four years ago. It still has more subscribers than we do, but freeCodeCamp YouTube channel has kind of taken on a life of its own and its getting like 500 million minutes of watch time a year.

[01:47:03] JM: So managing the open source project though, is there a much open source contribution? Because like there are a lot open source projects where like MongoDB, for example, like most of the contributions to MongoDB as I understand our from the company. There's nothing wrong with that. Open source is kind of like – It's just like a side feature of a lot of software products. It's a gesture? Open sourcing your software is a gesture, but it's not necessarily like open sourcing your software means you're Linux or Bitcoin and there's going to be like this tidal wave of people who are friendly contributors just dropping in and solving bugs that take them hours to dig through lines of code.

It's more like maybe you have somebody like maintaining the SSL library that's holding the Internet together, of you have like a person and a half that are keeping those libraries together. But it's really not like this avalanche of people that are just generously contributing. So it sounds to me like you're more like the kind of the MongoDB school of things where it's like –

[01:48:19] QL: I would argue that we are much more the opposite. Most of the contributions do come from the community. A lot of the big work is done internally just because there's less communication overhead and we can be more –

[01:48:35] JM: It's like big, heavy re-platforming work. The stuff that really takes like the deep work.

[01:48:40] QL: Yeah. Then we do occasionally have open source contributors who do significant deep work too. Just to give you some numbers, since I pulled it while you're talking. Just in the past week, October 22nd through the 29th, we've had 66 proposed pull requests. We merged 27 of them. Most of those have come from outside of the freeCodeCamp full-time staff. Same thing with lots of issues. There have been a total of 1,458 additions and 624 deletions to the codebase in the past week. If I look at the contributions, I can see that only about eight of those were from people working full-time out of the 30 commits pushed so far this week.

[01:49:24] JM: Okay. Interesting. One thing, as you're talking about all these different areas of the freeCodeCamp brand, you got not just the full curriculum that is an amazing product of its own. You're a media company. You've developed both a deep vertical expertise and teaching people to code, and a horizontal expansion that has been so well-executed that you have basically the biggest programming channel on YouTube. You have super active forums and content and chat rooms. I think your Slack channel was – I think you broke Slack, right? You literally like break the Slack free plan or something.

[01:50:16] QL: So we discovered a limitation within their infrastructure, that they can only have a certain number of people, and it was undocumented limitation. We hit it, and we had to quickly switch from their platform to Gitter, which didn't have that limitation. We still use Gitter. We still have contributors there, but we've shifted a lot of it over to the forum now.

The forum is great, because for every one person who's logged in, like writing answers to other people's question, there are 19 or 20 people who are just jumping in and from Google who are reading that exchange and benefiting from it.

[01:50:50] JM: That is sick.

[01:50:51] QL: Yeah, it's great. We're helping a lot of – When you go to the freeCodeCamp forum and you hang out there and you help answer people's questions, or when you ask a question, you're not just helping yourself. You're not just helping that person whom you're talking to. You're helping dozens of people. In some cases, some of these forum threads have had hundreds of thousands of views.

Yeah, you never know what problem that you're experiencing personally could be something that a whole lot of people are experiencing. By having that exchange right out there in the open in a way that's really easily indexed by Google, we are able to help a lot of people indirectly that way.

[01:51:25] JM: Let me ask you a question, because we both come from Quora. We both have deep affinity for Quora, and I feel almost like a debt of gratitude to Quora. That's how much I love that platform. But I almost wish that the software to build Quora was open source. I wish that so many things and that platform could just be like subset it out into their own products and turned into like highly verticalized things, because now the product is kind of – It's sprawling, and I wish it was more – I don't know. I wish it was like the old days, which is what everybody says about every Internet product that they used to love.

Wouldn't it be cool if you had your own like – Instead of – Discourse sounds great. But it would be better if it was Quora, right?

[01:52:22] QL: It's not really a fair comparison, because the guy who founded Quora, Adam D'Angelo, he is the closest thing to a software engineering genius that I know of before he created Quora and took hundreds of millions of dollars of his own money and funded it. He was the CTO at Facebook, and he just had so many incredible insights that I think we're missing from Stack Overflow, and is certainly missing from Yahoo Answers and other sites that he was competing with. That platform is just remarkably well-built and well-thought out and it works amazing.

So I have nothing but good things to say about the way Quora is built. I think it would be a very tall order for a team even with the head of Stack Overflow helming it to create something as refined as Quora. That said, the spirit of Quora, which is asking and answering questions and all that, that can work on any platform if you have the community of thoughtful people who are willing to turn around and show their expertise and their insights with one another.

So the tools themselves – You could give Quora, like you could take the existing codebase and give that to kind of a community that's marginally active and they wouldn't necessarily be able to do that much with it. But you can give PHPBB or some really old form tool to a vibrant community of people who are really passionate about something and they would build up a huge, huge ecosystem of questions and answers and people helping one another and come up with these great forum admins and forum moderators and all that stuff. That stuff would just organically bloom.

So if anything, if freeCodeCamp has succeeded, I would say like the forum, the reason it's succeeding is because we have that community, we have these super passionate moderators and people who genuinely love helping other people that have come at it. That's why Quora has been such a huge success too, and that's why you and I were out there, because we both felt a big affinity for one another and for other people who were interested in software engineering and we enjoyed answering their questions there.

[01:54:35] JM: Do you ever feel like the sprawl of things that you are doing with freeCodeCamp, does it ever feel like some of it is a distraction? Maybe if you're time and dollars and focus was reallocated exclusively to the curriculum, you would be having measurably better output?

[01:55:00] QL: So there are diminishing returns to what you can get from focusing on different areas. I would argue that the publication and the forum and the curriculum, they are complementary to one another and they buttress one another. The publication is a way of keeping people engaged when they don't have a keyboard handy. They are just eating lunch and they just need to be able to swipe through something and read something on their phone. They're not ready to dive into a deep in-depth coding session.

The forum is there to help people who encounter problems or have questions while they're working through the curriculum. On every page of the critical, there's a button that you can press that opens up a pre-populated form post with the current state of your code and everything in it and it makes it very easy to ask a question. So they complement one another.

Now, things that don't necessarily complement one another that we've pursued nonetheless, I would argue, would be Code Radio. FreeCodeCamp has a radio station, an Internet radio station. If you go to coderadio.freecodecamp.org, you can listen to 24/7 music. That came out of an experiment I did on YouTube where I decided like, “Hey, let's create music that we'd like to listen to while we’re coding.”

We had this music producer from Los Angeles named Lawrence Yeo and he decided to help curate a big-play list of thousand – I think 1500-ish tracks of hip-hop beats, and we put those together and it was really successful. We’d have tons of people watching on YouTube.

Now, there was some anime clip, and one of the songs it was like a, “Haya,” or something from anime, some Japanese company that had – It was a media holding company. They probably just bought some random anime series back in the 80s and were like using their blanket copyright detection tool. They flagged it and YouTube immediately shut down the channel.

So we get to move it. So we’ve moved it over. We found a really awesome open source Internet radio tool called AzuraCast, and we were able to within about 48 hours get the music back up. So that's something that like we don't need a radio station, right? Coding community doesn't need its own hip-hop beats radio station, but it's really nice and it didn't take that much work. It's not a whole lot of maintenance work. We’re getting ready to launch a classical version where we just play orchestral music. So people can just put on their headphones and listen to some Bach or some Vivaldi.

We’re going to have a lot of people listening to that too, but that's one of the things that people use a lot. At any given time, we might have 200, 300 people concurrently listening to Code Radio. So it's something that doesn't take a whole lot of effort, but it’s used by a lot of people. So when we see opportunities like that, yeah, we can do that. I don't think is a huge distraction.

Elon musk runs Tesla and he runs SpaceX and he runs Solar City and he runs like the whole Hyperloop thing, right? It's difficult to see the synergies in all those, but he might be able to rattle off some synergies. What I think we’re doing is dramatically more conservative than what he is doing. Not that that’s necessarily the best example, but –

[01:58:17] JM: Yeah, I think you forgot the brain company. Yeah, I think you forgot the brain company and the tunnel company. There's a lot of stuff. I actually think this is one way in which the strong narrative of the financing of software companies has maybe been to the detriment of software innovation in some ways.

I'm generally a big fan of the financing and the capital structures that have enabled software companies to flourish, but one thing that I think might be a little bit outdated is the notion of rigid focus even on company – Even in companies that are like super early, like I get it, that like relentless focus can be really good for developing that first product market fit and whatnot.

But I also think that like this one thing I loved about Amazon, was it was just like you want to start some half-baked Amazon product, like go do it. You're there. You've got access to these resources. You've got access to people with like crazy ideas. You want to go start like a grocery thing, or like some weird storage unit products. You can probably marshal the resources if you put your mind to it.

I think like if you're in your position when you're running freeCodeCamp, again, if you're trying to be inspirational, one way to be inspirational is like cultivate a sense of like we say yes to things. Should freeCodeCamp have a radio station? Why not? It doesn't cost us anything. Ting the community likes it. It makes people feel good. Yeah, why not? Then when you have that attitude of just like saying yes to kind of random stuff, you open the door to innovation, because people start to feel comfortable like, "Okay. I can express my creative ideas. Many of my creative ideas do not have a financial outcome." They do not have an outcome of helping people learn to code faster, but we've got this quirky community. Yeah, why not spend do you know .001% of the budget or the time budget just making things a little bit more musical?

[02:00:37] QL: Yeah, and we think other experiments too because precisely of what you said. It makes the community seem more fun and more adventurous. You're just taking low stakes bets. I mean, worst-case scenario with the Internet radio station would've been a few hours of time. Maybe dozens of hours, or hundreds of hours spent in the wrong direction. But that kind of experimentation can lead to big successes. I'd say Code Radio has been up a big success. It's been a thousand fold return on initial effort in terms of people benefiting from it. If you took the time that's been spent on Code Radio versus the time people spent enjoying the jolts of Code Radio.

Programmer playing cards is another thing that we did. So we went through – And I'm a big history buff, and I love learning the technology history especially. I went through and came up with a list of all a bunch of people who I thought were really inspiring programmers and we printed these playing cards. So every playing card has a different famous developer from history. I mean, aside from [inaudible 02:01:43], all the people are from the 20th century. Some of them were born in the 20th century and became famous in the 21st century. For example, like DHH, the creator Ruby on Rails, or Satoshi Nakamoto, people like that.

But it's a great way for you to be playing cards, playing poker, playing whatever traditional card-based game you want to play. While you're sitting there waiting for someone to decide if they want to call your bluff, you can be reading the accomplishments of, for example, Linus Torvalds, or some of the more contemporary developers. So it's just a little thing that we did that we thought would be fun and we were able to do it and it was profitable in the sense that I think we spent several thousand dollar doing the design work and getting the cards printed and it paid for itself.

Now, wasn't like some huge smash profit center for freeCodeCamp, but it was a cool thing to do for the community and a lot of people took pictures and we go to events and people have programmer playing cards, and it's a cool thing. It's a cool instructional tool. So freeCodeCamp doesn't shy from doing those things even though we're fairly limited with resources. If we think that it's a small bet that has a large potential payoff.

[02:02:59] JM: You have a family and you run an extremely successful web platform with very few full-time staff that you can delegate stuff to. You get a shoestring budget. I don't want this to turn into a lifestyle advice podcast. But I do know that there are an increasing number of people who work in a similar fashion to you. I think I am one of them. I don't have a family, but I run my own weird business, that I have one person I work with full-time, Erika, and it's awesome to have some teamwork.

But, generally, it's kind of a lonely – I don't want to say lonely, but there's like Indie Hackers and there's like advice books and stuff, but this kind of business is weird and there's a growing number of people who are managing this kind of business, or working it, even just working as a contractor these days, like an online digital nomad contractor juggling two different jobs and then working on your own side projects. It's a very 21st-century existence.

Give me some subtle tips about how to manage this kind of work.

[02:04:23] QL: Well, I spent a lot of time talking to people through my computer, mostly through email, sometimes through Google Hangouts calls. I do a few of those every day. I don't really see many people in-person. I have friends that I talk to periodically, but it's rare that I meet up with them, like maybe once a month would be like, "Hey, I'm going to hang out with friend and we're going to play a board game together or we're going to go to the gym together," right? Most of it is pretty solitary, but that's good for me, because I just like to read and I like to just go for runs and think.

A big chunk of my life is just me sitting there and thinking or me walking with my kid on my shoulders walking down the street, taking them to the parker or taking my daughter to school in the morning. It just gives me a nice rhythm to life to think, and I've got a very regimented routine. I always wake up the exact same time. I always go to sleep at the same time. I generally eat just a few meals over and over. So there's not a lot of variety, but that monotony that I built up, that routine, provides a very nice framework for me to explore individual things that I'm interested in without the distraction of having to make a lot of decisions throughout the day. I wear almost exactly the same clothes every day.

Just those little pieces of routine keep me on board. As soon as I have to get on the plane and fly somewhere, it is extremely disruptive. Probably more for me than for a lot of people, because I don't have that family. I don't have that routine to a whole new place. But that's just me personally. I really enjoy it. I enjoy, "Okay. I've got this block of time at the end of the day. Do I want to watch an episode of The Expanse, or do I want to go for a long walk and listen to this audiobook? Do I just want to spend it –

[02:06:18] JM: Audiobook?

[02:06:18] QL: Yeah, or podcasts. That's one thing that I do to keep myself company, is I listen to a huge volume of audiobooks. Every day I listen to like The Economist podcast. I listen to the New York Times podcasts, the Washington Post Podcasts, like all the daily news podcasts just to keep on top of the world events and things like that. Then I listen to, for example, your podcast. I listen to you probably several times a week. I listen to Indie Hackers. I listen the freeCodeCamp Podcast, which increasingly is hosted by Abby. I'll listen to my episodes too just to see how I performed on them and kind of evaluate how good I wasn't of asking questions, things like that.

But I just take a ton of information in my information diet. I don't have any apps on my phone. I just use my browser. I use like Firefox Quantum, or Firefox – It's the one that isn't completely privacy-focused and I can just – I just spend a lot of time like Wikipedia, or I spend a lot of time like just Googling different things.

[02:07:16] JM: Wait. So you don't have Audible or a podcast app on your phone?

[02:07:20] QL: So I have the built-in apps that came with the iPhone. So I do have the podcast. I just use the stock Apple podcasts app.

[02:07:26] JM: And for audiobooks?

[02:07:28] QL: For audiobooks I use Libby, which is like the library system.

[02:07:32] JM: Really?

[02:07:34] QL: Depending on which city you live, you may be able to use one of many different library apps, and basically you just check out the virtual good, if you will. There's like this for scarcity, this artificial scarcity. But like I have a whole bunch of audiobooks that I have on hold. So as soon as somebody else returns them, I can listen to them, right? But it's clean, because I try to save money. That's one thing that we didn't really get into, but the reason why freeCodeCamp able to grow as much as it did despite the fact that it took us two years to get tax-exempt status. Almost three years. It took us a whole lot of time before we could actually start asking for donations, things like that. I spent more than $150,000 of my own money keeping freeCodeCamp afloat during the first few years.

The reason I even had that money to begin with was because my wife and I are like compulsive savers and we were saving to buy a house, and it's just pretty much every decision is made to minimize the amount of money. That's just a holder for me spending a lot of time overseas in China. I saw how frugally people are living, and I thought like, "Wow! That $60 that I was going to spend on my videogame, that somebody's – That's how they survive for the entire month."

I met tons of people who were living off of $5, $10 a day and interacting with them and they were going to school. I met people who were going to go work at the state-owned enterprise, government banks and stuff like that and they were just carrying their thermos filled with tea and eating there home- prepared buns for lunch and just being extremely economical. I took a lot of inspiration in that. So I just try to save as much money as possible.

One of my favorite restaurants – This is going to make me sound like a crazy person, but I just go to Sam's Club, and they've got this hotdog-drink combination for like $1.50. Then I'll get like a pretzel, and my entire lunch is like $2.50. I could sit down. They've got good Wi-Fi and I could just eat a hotdog and eat a pretzel and drink a diet drink and work. It costs almost nothing, right? Those little opportunities to save money, those do matter when you're working as a teacher and you're trying to accumulate enough money to buy a house, or as it turns out, finance an NGO.

[02:09:50] JM: Man, there's a lot a lot in there that we could dig into further. I really wish you were in town so we could go running more often. I miss running with you. Like you, much of my mental processing and like preparation for shows really occurs when I'm like running outside, because you're running outside and you're seeing people, and you're seeing restaurants, and you're seeing the world develop. You're feeling the temperature changes. You're feeling the passage of time.

Every year, you feel technology, like calcifying it, some kind of essence in that passage of time. You feel the technology. You see more people on your runs interacting with technology in different ways, "Oh! There's more people with AirPods, and look at how that – Look at how my dad uses a smartphone." I love running as a daily routine of interacting with the world, of engaging with my surroundings. I jot down a lot of notes when I'm running, and it's all the better when I'm running with somebody else.

I actually started doing this. Maybe we could do this sometime, this thing called virtual runs. A term I coined, where basically like you just put on Bluetooth headphones and put your smartphone in a fanny pack, pack unless you're like a cool person with a smartwatch who doesn't need a fanny pack for their smartphone. But you can just run with like headphones in and you can have a virtual run, and it's like you're running with Quincy Larson, but you're just talking to him, and perhaps he's running, or perhaps he's washing dishes. This is what will supplant the podcast, is my theory, the virtual run, the interactive podcast. Anyway, I really miss hanging out with you, man, but we never did that often, but it was fun.

[02:11:47] QL: Yeah, now that I have kids, I never hang out with anybody very often other than my kids. Having kids is a pretty major life change.

[02:11:55] JM: Totally fair. Totally fair. Not trying to guilt trip you or anything. Just we had some great – I mean, we haven't even gotten to get into politics in this conversation.

[02:12:06] QL: Probably for the better.

[02:12:08] JM: Probably for the better. Okay, just one or two questions. How's the world going to change with broadly accessible online coding education?

[02:12:20] QL: First of all, I'm optimistic that it will mean fewer people going into debt to pay for their college education. I would not encourage anyone to go into debt even to attend like a fairly prestigious university. I'd really encourage them to go to Community College. If they have the resources, sure, go to a really solid-state school right off the bat. But see if you can save some money by going to Community College and transfer in to a better, like a bigger state school. See if you can figure out ways where you can be working while you're going to school, doing like an internship.

A lot of software development internships pay really well. You can potentially pay for your rent and your school with a good internship. So I'm optimistic that freeCodeCamp can provide a way that people can set themselves up to get these internships while they're still in school. I'm also optimistic that a lot of fewer people will need to quit their job in order to transition into a new field. Certainly, more technical fields, like software engineering.

So on a whole, I think it will save people a lot of money and it will also give more flexibility, and that's what it's all about at the end of the day, is saving people from having to make a bunch of lifestyle compromises in order to be able to go ahead and be gainfully employed.

[02:13:33] JM: Describe the software engineering media landscape as you see it.

[02:13:38] QL: Sure. There are some awesome technical journals that are inaccessible for a lot of people. They just assume you know a whole lot of stuff. If you're tenacious, you can read them and you could just constantly open up Wikipedia or you can open up your dictionary or whatever tool and look up terms as they come out of the text at you while you're trying to read these papers. There are also a lot of great blogs that discuss a lot of technological developments. There are open platforms where pretty much anybody can publish articles, like dev.to, and Hash Note, and Hacker Noon.

Then there are tech publications, some of which are now paywalled increasingly, like I was looking at Wired Magazine and a bunch of other ones yesterday and I noticed that most of them have up a paywall at some point. Maybe they'll let you see a few articles and then you're hitting a paywall. So those are becoming less open, which is unfortunate, but I can understand that it costs a lot of money to have a newsroom and run those.

Then there is freeCodeCamp, which is kind of a hybrid between a traditional tech publication and a blog, and we edit everything that gets published on our publication. We are very selective about whom we let be an author. Generally, 1 in 15 people who apply will get authorship. Then they'll be able to write articles and we edit those articles and we publicize those through our social media and also through my email blasts and LinkedIn groups, or our LinkedIn alumni association, a lot of places.

So it's common for people to publish articles and get tens of thousands of views on them. Not every article does that well, but a lot of them do. If you spend a lot of time writing an article, we're going to heavily publicize it. Not just right after it gets published, but we continue to publicize things for months and years after they've been published if they're still relevant and if they're kind of more explanatory journalism that explains fundamental concepts.

Just to backup. I would say there are open platforms where anybody can publish anything about technology. There are technical blogs run by individuals. There are publications, like Wired, The Verge. Then there are sites like freeCodeCamp that are publications.

[02:16:07] JM: Last question. If you weren't working on freeCodeCamp, what would you be building?

[02:16:14] QL: I'm not sure. It will probably just be another technology education tool. There's a possibility that I'd be doing English language education. I think English language education is vital and it's a market that is relatively inefficient in my humble opinion. It's a massive market. FreeCodeCamp may very well eventually teach English, which sounds like a huge departure from what we do. But I think it goes hand-in-hand with helping people get 21st century jobs.

English is now the language of business. It's the language of science, and increasingly it's kind of becoming the language of culture. Now, there are certainly a lot of other languages that are used in culture, Japanese animation and Korean in pop music and things like that. But I think that English is still kind of the standout. If you wanted to make a movie or if you wanted to make an album or if you wanted to write a book and have a wide audience, English would be the natural choice. So I would probably – If I wasn't doing technology education, create some sort of project to run English language education.

[02:17:22] JM: Quincy, any closing thoughts about what you'd like the listeners to hear?

[02:17:27] QL: Well, I said earlier that I'm really bad about getting people to donate to freeCodeCam. So my closing statement would be if you're listening to this and if you've gotten some value out of freeCodeCamp over the years and if you've listened this far into this interview, thanks for listening to me, and in Jeff's excellent questions. I would encourage you to consider donating to freeCodeCamp. You can set up a $5 dollar a month recurring donation, and that is hugely valuable to us, because it gives us the ability the project what are our budget is going to be.

Monthly recurring donations are by far the best. But if you just want to give a large lump sum, like Shawn Wang did or some of these other graduates from freeCodeCamp have been doing, we would welcome that too. You can use the @payitbackwards, and a lot of people will see that, and I'll almost certainly re-tweet it as well.

So we would very much appreciate your support. FreeCodeCamp can efficiently deploy three or four times as much capital as we have currently. I am confident that we could continue to be just as effective delivering 50 hours or more learning to people around the world per dollar. We can continue to scale up. We're still very early days.

[02:18:35] JM: Quincy Larson, thank you so much for your generous expansive time.

[02:18:42] QL: Thank you so much for coming up with these thoughtful questions, Jeff. It's been great talking to you.

[END OF INTERVIEW]

[02:18:54] JM: If you want to extract value from your data, it can be difficult especially for nontechnical, non-analyst users. As software builders, you have this unique opportunity to unlock the value of your data to users through your product or your service.

Jaspersoft offers embeddable reports, dashboards and data visualizations that developers love. Give your users intuitive access to data in the ideal place for them to take action within your application. To check out a sample application with embedded analytics, go to softwareengineeringdaily.com/jaspersoft. You can find out how easy it is to embed reporting and analytics into your application. Jaspersoft is great for admin dashboards or for helping your customers make data-driven decisions within your product, because it's not just your company that wants analytics. It's also your customers.

In an upcoming episode of Software Engineering Daily, we will talk to TIBCO about visualizing data inside apps based on modern frontend libraries like React, Angular, and VueJS. In the meantime, check out Jaspersoft for yourself at softwareengineering.com/jaspersoft.

Thanks to TIBCO for being a sponsor of Software Engineering Daily.

[END]