EPISODE 883

[INTRODUCTION]

[0:00:00.3] JM: A hackathon is an organized event where participants work together to build a product or a tool. Hackathons are about creativity and learning and exploration. A developer that is participating in a hackathon is often working on something that is outside of their normal day-to-day focus. Hackathons can provide significant value to the participants. Hackathons have led to friendships and new companies and newly developed confidence that can be critical to a developer who feels uncertain in their ability to launch their own projects.

Jonathan Gottfried is a Co-Founder of Major League Hacking, an official student hackathon league that powers invention competitions. Major League Hacking is a B corporation that is focused on improving the education and community of technology leaders and entrepreneurs and young hackers. Jon joins the show to discuss hackathons and how he's built and scaled an organization that's devoted to creating systematically successful hackathon experiences.

Podshees is an open source podcast hosting platform that we recently launched. We're building Podsheets with the learnings from Software Engineering Daily. Our goal is to be the best place to host and monetize your podcast. If you've been thinking about starting a podcast, you can check out podsheets.com.

FindCollabs is the company I'm working on. It's a place to find collaborators and build projects. We recently launched github integrations. It's easier than ever to find collaborators for your open source projects. If you're looking for someone to start a project with, FindCollabs has topic chat rooms that allow you to find other people who are interested in a particular technology, so that you can find people who are interested about cryptocurrencies, or React, or Kubernetes, or mobile development, whatever you want to build a product with.

With that, let's get on to today's show.

[SPONSOR MESSAGE]

[00:02:06] JM: You probably do not enjoy searching for a job. Engineers don't like sacrificing their time to do phone screens and we don't like doing whiteboard problems and working on tedious take-home projects. Everyone knows the software hiring process is not perfect, but what's the alternative? Triplebyte is the alternative. Triplebyte is a platform for finding a great software job faster.

Triplebyte works with 400-plus tech companies, including Dropbox, Adobe, Coursera and Cruise Automation. Triplebyte improves the hiring process by saving you time and fast-tracking you to final interviews. At triplebyte.com/sedaily, you can start your process by taking a quiz. After the quiz, you get interviewed by Triplebyte if you pass that quiz. If you pass that interview, you make it straight to multiple on-site interviews. If you take a job, you get an additional $1,000 signing bonus from Triplebyte, because you used the link triplebyte.com/sedaily.

That $1,000 is nice, but you might be making much more, since those multiple on-site interviews would put you in a great position to potentially get multiple offers. Then you could figure out what your salary actually should be.

Triplebyte does not look at candidates' backgrounds, like resumes and where they've worked and where they went to school. Triplebyte only cares about whether someone can code. I'm a huge fan of that aspect of their model. This means that they work with lots of people from non-traditional and unusual backgrounds.

To get started, just go to triplebyte.com/sedaily and take a quiz to get started. There's very little risk and you might find yourself in a great position getting multiple on-site interviews from just one quiz and a Triplebyte interview. Go to triplebyte.com/sedaily to try it out. Thank you to Triplebyte.

[INTERVIEW]

[00:04:25] JM: Jon Gottfried, welcome to Software Engineering Daily.

[00:04:27] JG: Yeah, Jeff. Thanks for having me.

[00:04:30] JM: What is the ideal model for programming education?

[00:04:33] JG: That's a great question. Programmers learn best by doing. When you go off into the world, computer science is one of the only professions where you're essentially paid to learn as you go. That not all applies before you enter the industry too. When you look at who the most successful programmers are, it's people who have been extremely prolific outside of a traditional classroom environment.

You hear about all of these college dropouts who start working on their projects. You hear about people doing boot camps, you hear about people going to hackathons. Those folks who are seeking out opportunities to build actual real, working products are the most skilled programmers entering the industry right now.

[00:05:17] JM: Just to dig a little bit deeper, how would that be reflected in an educational curriculum?

[00:05:26] JG: It's really difficult to reflect in an educational curriculum. Curriculum is often designed around testing specific knowledge of concepts, skills, theory. It's fairly hard to test something abstract, like how well did you design the architecture of this system, right? Or how useful is the product that you've created?

That's really hard to grade at a core level, and so you don't see it in a lot of curriculum. What you do see is a lot of projects within a very narrow scope or framework. It's up to the students to figure out how to actually build beyond that. Some of the most novel curriculum does involve a lot of project-based, self-led learning. It's really not part of the majority of CS classes.

[00:06:16] JM: How do hackathons fit into a programmer's education in an ideal world?

[00:06:23] JG: Student hackathons are an environment where students can actually seek out those self-driven learning experiences, rather than sitting around on the weekend and doing a homework sheet, or a quiz, they're going out and coming up with their own project, building a working version of it with people that have a wide variety of skill sets, and then actually putting it

out there in the world to get feedback from their peers and from mentors in the industry. That's super powerful.

There's actually a lot of educational pedagogy that goes along with that. There have been a number of studies now about the values of learning from your peers, rather than super experienced people. There's a lot of research out there around project-based learning and the environment that you need to be in to learn skills, and hackathons bring all of that together in this really special bubble, where all you're doing is building. That's a really powerful thing. Usually, the first time a student goes into that environment, it changes their whole perspective on the world. It's like a switch that's flipped.

[00:07:34] JM: How does it change? What changes about their perspective?

[00:07:37] JG: I describe it like opening a black box. Oftentimes, before someone has gone to a hackathon and a similar format events, they've never really gone outside of the confines of what they were asked to learn or build, right? Even if you're in a job, a lot of the time you're given a scope and you just have to build to a scope. In class, you're learning the specific topics and theories that you need for your test.

Hackathons are often the first place where someone is exposed to the idea that they can do whatever they want. That's super powerful, especially for programmers because being able to build software is one of those rare skills that in a very short period of time, someone can build something that could be accessed by anyone in the world, right? That's functional. That is technically complex and interesting. That causes people to realize that they have this magical power almost. It's like going to Hogwarts for the first time. It really just changes your perspective on what you're capable of and it often forces people into this mindset of, "Okay, now I can just learn whatever I want, whenever I want."

[00:08:57] JM: My sense is that the way that the computer science curriculum has developed is largely due to the fact that in the earlier days, programming was much harder. You had much lower leverage as an individual programmer. The way that the curriculum is structured reflects that low leverage as a developer. Is that your sense as well? Why has the curriculum of the

computer science university developed in a way that feels so alien to people who are actually building modern software projects?

[00:09:42] JG: I think you're absolutely right, that a huge part of it is a result of the previous complexity of building software, right? Even 15, 20 years ago, which isn't that long in the grand scheme of things, you had to have a much deeper understanding of math to be able to be a good programmer, because your resources were more constrained. You had to worry about memory management. You had to worry about CPU cycles.

If you're writing a node app and deploying it to some cloud server, you're not really concerned with that anymore. You have a lot more resources at your disposal for a pretty low price. I think that's a big part of it. Curriculum is just slow to keep up with that thing. I think the other part of it has to do with the culture of universities. There's a really great book that I love about hacker culture, called *Hackers* by Steven Levy. It talks about the early days of hacker communities at MIT, Stanford, a lot of these other top tier schools.

What you start to realize is that even way back in the 1950s, right? When computers were really just starting to come about, you have this culture of let me just figure it out. Let me try something. Let me see what works. See what sticks. That's the exact same hacker culture we talk about now at hackathons. The problem is that culture didn't really spread beyond MIT, Stanford, top-tier universities. For decades, it was siloed on these campuses.

I honestly think that's part of why they maintain that status as top-tier universities, is because beyond their strong research and academic curriculum, they had this hacker culture that was ingrained into their school's DNA. I think that the proliferation of hacker clubs and hackathons and extracurricular learning beyond those campuses has created that culture in a lot more places. People are starting to get the same benefits from it now that hackers at MIT did 50 years ago. I think that that is rapidly changing how people learn programming.

[00:12:00] JM: The structure of a hackathon is dissimilar from how a software engineer is typically working. The way a software engineer is typically working is in a quiet room, oftentimes by themselves. They're just solving a problem and they're spending eight hours on this small problem that is a subset of a larger application. A hackathon is a more condensed, frenetically-

paced environment, where a programmer is working around a large number of other people. Why is it that a hackathon is useful, despite the fact that it's so different than how a programmer is typically working in a productive context?

[00:12:51] JG: For one, you're not building as part of a larger team or project. The constraints are a little bit different. I think the way you described it is pretty accurate, right? It's a little frenetic. It's a little crazy. When you go to a hackathon that has a thousand people sitting in a room coding, it's not a silent environment. It's almost somewhere between a party and a classroom. Obviously, people aren't binge drinking and hanging out. At the hackathon, it's a much more in a professional environment than that, but they are having fun.

For a lot of the students who go, it's a social experience as much as it is educational or professional. You can think of it like a microcosm of what it means to code. When you're working on a project and you get into that flow state where you're really thinking clearly and producing good work and structuring your time and effort effectively and it feels really good, a hackathon is a bite-sized chunk of that.

What you're producing at the end is not a feature on some app that already exists, it's a small prototype. It's not meant to replicate good architecture or engineering practice. It's meant to replicate the process that someone needs to go through to build an MVP, or try a new technology for the first time, or show someone that crazy idea they've been thinking about for six months. I think that's a common point of confusion. A lot of people are like, hackathons don't produce anything real. Nothing comes out of them.

I would actually argue that that is exactly the point. It reduces the risk associated with building something, if you don't have to worry as much about the value of the output. It allows people to be more creative, because they can take more creative liberties, right? They can build something that doesn't have to show immediate value. I think that's super valuable from a learning and also just a creative satisfaction perspective.

[00:14:56] JM: That's a very important point. The idea that by working on things that may have no practical application, or may not even make it to the minimum viable product that you plan to build to, you can nonetheless make progress as a programmer, make progress in your career.

Why is that? Why is it important to work on things that basically hit a dead end, that don't have any productive application?

[00:15:28] JG: I compare it to how artists learn. Imagine if you're a painter, and someone is like, “Oh, go paint the Mona Lisa.” The Mona Lisa was not the first portrait he ever painted. There were a hundreds of portraits before that that no one's ever seen, because they probably got thrown away, or painted over, or just lost to time. I think programming is a really similar skill set in that, it is both a technical and creative discipline. There's going to be a lot of trial and error along the way to becoming great.

You're not going to produce a masterpiece, nor you be expected to the first time around. When we think about why that's beneficial, most projects that people work on as programmers, you're either being graded on for class, or you're being evaluated on as part of your work performance. That forces you to think about things in a different way that's probably more narrow, right? You're not necessarily going to take a huge risk with a new technology at work, because that could end really badly for your company. That's probably a good thing, right? We don't want all the programmers out there just throwing whatever new libraries is like, cool into production, because that's not good architecture practice.

When you're learning something for the first time, that's fun, that's interesting, that's cool. It allows you to learn new concepts that you wouldn't necessarily encounter on a day-to-day basis, because there's fewer constraints. For example, personally, I did a lot of the early hardware development I ever encountered at hackathons. I never worked in hardware. No one would ever hire me to work in hardware. I don't have an electrical engineering background.

I was really intrigued by it. I wanted to learn about it. Given that it wasn't something that existed in class, it wasn't something that existed at work, the first place I tried it was through hackathons and these extracurricular activities and I loved it. It's something I got really excited and passionate about.

[00:17:38] JM: Why was the hackathon environment useful for you to explore that subject? Why wouldn’t you just do that by yourself at home?

[00:17:46] JG: I actually think this comes back to the types of people who are attracted to programming and frankly, the diversity of the industry. It's not a safe assumption that everyone who is a programmer likes to work alone in the dark in quiet. That's a stereotype. It's a cliché that goes back a very long way. I don't think it's accurate and I don't think it reflects the next generation of people entering that industry.

I really think that hackathons, given that they are both a social and professional environment, give people opportunities to experience the tech world in a different way that's not normally represented in the mainstream. For me, as someone who's fairly extrovert and outgoing and I like being around people, going to a hackathon and learning those skills is encouraging, right? Because I can sit there, everyone around me is also working on projects. That's a really mutually beneficial environment to be in, because there's energy that comes from being around other people who are doing cool stuff.

Part of it is I can literally raise my hand and be like, “Hey, who here knows how to use Arduino?” Someone will come help me. The other part of it is it's just more fun to be around people when you're working on something.

[00:19:10] JM: Let's say I'm showing up to my first hackathon. What should my expectations be and how should I optimize for that experience?

[00:19:22] JG: It depends on your skill level. If you're going to your first hackathon and you're brand new to programming, I would approach it with as open a mind as possible and really, try to join a team of people who are slightly more advanced than you. Because you'll get to contribute to what they're working on and almost pair program to learn the skills that they're already familiar with. That'll be the main benefit you get from it. I actually think hackathons are a great introduction to programming, if you're surrounded by teammates who have slightly more advanced skills.

I think that if you're more advanced and want to go a little out there, outside of your creative box, I usually to talk to people about what their ideas are. I find that that gets the creative juices flowing a little bit when I hear, “Oh, this person's working on a music tech hack. This person is working on hardware.

This person is doing something wild with biotechnology, analyzing blood samples that they found on Google Image." There's a lot of really weird stuff that you see at hackathons. For me, as someone who's been through a lot, my favorite thing is here where everyone's working on, see if any of that strikes a nerve and I want to work on it with them. Then maybe if it doesn't, it'll inspire a different idea that I've had sitting in the back of my head.

I think that the folks who go to a ton of these events, often just keep a running log of projects. They really have a backlog of things they want to try or play with, or they've been thinking about, and hackathons are their opportunity away from school and work to try that. It really does apply for people of all skill levels. It's a different experience depending on where you are in your journey.

[SPONSOR MESSAGE]

[00:21:18] JM: Apache Kafka has changed the world of data infrastructure. Kafka Summit is the place to learn about new design patterns and engineering practices in the world of Kafka. Kafka Summit returns to San Francisco, September 30th through October 1st, 2019. Kafka summit has sold out in New York and London. The San Francisco event is likely to be just as popular.

Listeners of Software Engineering Daily can get 25% off their ticket to Kafka Summit by entering promo code SED. With the promo code, Kafka Summit is only about $900 to attend. If that's still too expensive, you can consider asking your company, or your manager to pay for your ticket.

Kafka Summit is an educational experience with top engineers from places like Netflix, Microsoft, Lyft and Tesla. At Kafka Summit, you can meet with experts who will help you address your toughest Apache Kafka and event streaming questions. Or you can start to learn the basics of how to deploy and operate Apache Kafka. There are also hands-on beginner and advanced training courses available, as well as certification.

Join the Kafka Summit, September 30th through October 1st, 2019 and get 25% off your ticket by using promo code SED. I plan on attending Kafka Summit and I hope to see you there.

[INTERVIEW CONTINUED]

[00:22:58] JM: There are some hackathon environments where the expectation is this is going to be really long – I mean, hackathon. It sounds very long. I'm going to be exhausted by the end of it. My stomach is going to hurt from the strange foods I've been eating. Maybe there's this element of competition. Maybe I don't like competition. What are the anti-patterns of hackathons, or are all those things I listed just features that some people don't like, but many people do like?

[00:23:29] JG: It's a little of both. I think as hackathon organizers, it's important to think about the different needs that attendees will have. You can't assume that everyone wants to stay up all night. You can't assume that everyone wants to drink energy drinks and eat junk food for 24 hours. That's terrible. Frankly, if I go to a hackathon where that's all they have available, I'm leaving to get food. That takes you out of the bubble.

I think, it is really important as event organizers to accommodate a lot of different needs. Newer hackathons have more resources like that. Most student hackathons have a sleeping room, where you can go take a nap overnight. Most hackathons have salad available. Things that are totally against the cliché of what hackathons are. I think that's how you identify is this a good hackathon, right? Has someone put thought into all of those elements to make it a better, or more inclusive environment?

I think that when it comes to the competition element, that does attract some people. I don't think that hackathons are – it's not a super competitive, crazy environment. I almost compare it more to a marathon. When you're running a marathon, which to be fair I've never done in my life, you're competing against yourself. There's a very small percentage of people who can actually win a marathon, but most of the people there want to see if they can finish, right? What their time is. Maybe they beat their previous time.

Hackathons are the same way. Everyone's relative version of success looks different. I think that's a good part of it. It's okay that some people there are building their first web app ever and other people there want to win. Those things can coexist, as long as the people who want to win are participating in a way that is collaborative and helpful to the people who are brand new.

I actually think that in my experience, a lot of the folks who are super competitive are often some of the best mentors, because they're really familiar with the environment, they have a lot of experience and they want to spread the experience that they've had so positively to new people. They tend to invest a lot of time in helping folks who are less familiar with hackathons.

[00:25:53] JM: You started Major League Hacking. Why was the idea of the hackathon a compelling enough venture to go after, that you would build an entire business around hackathons?

[00:26:12] JG: As with most businesses, it wasn't quite that well thought out in the early days. Frankly, now we do go beyond hackathons. Really, the thing that drew us in at day zero, right? When we quit our jobs with no money to work on this, was our own personal experience with the format and with the communities. I had been a developer evangelist at Twilio.

My co-founder, Swift, had been a developer evangelist at SendGrid. He was in the process of selling his previous company. He quit his job at SendGrid. He started working on MLH for maybe four to six months before I joined it full-time.

Essentially, what happened was there was this small handful of student hackathons that had been around for a couple years. I'm talking five events. We would go to those events as mentors, or sponsors, or just to help out. It was one of those a little lightbulb moments, where you sit there and you're like, this is fundamentally changing these students' lives and also changing the environment that they exist within. People were leaving those events and being like, "Wow, I want this on my campus too."

It had this viral spread. That when we actually did start working on MLH in earnest, basically in spring 2014, we went from five events to 40 events overnight. Because all of these people had seen this, they had experienced it. Suddenly, they wanted to do it too. They were coming to us for advice, because we had been in other mentors at these events.

[00:27:55] JM: Attendees. Attendees from your previous events would be like, "We want this. We want more of this. How do we make it?"

[00:28:01] JG: Yeah. Some of that was frankly, competitive. Some of that was like, “Oh, I want my university to stick out more than this other university.” There were some fun rivalries in the early days. For a while, there was a rivalry between UPenn and University of Michigan to see who had the biggest best hackathon. There were definitely some downsides of that, but it drove a lot of early interest.

[00:28:24] JM: Like what? What are the downsides?

[00:28:25] JG: I mean, you don't really want people competing to have the biggest event. That's not objectively a good thing. It resulted in some really cool stuff. I remember one of the first M hacks at University of Michigan, they managed to hold it in the big house, which if you're a college football fan, is one of the largest stadiums in North America. They had a thousand hackers in the luxury boxes at the stadium. Then in the middle of the night, they let everyone out to the field to hang out and play frisbee and have a good time outside of the hackathon. That was awesome, right?

We were shooting off a t-shirt cannon with hackathon t-shirts. That's an unreal experience. That was fueled partially by their desire to be bigger and better than Penn's hackathon, Penn apps. There was definitely some rivalry that fueled a lot of the growth. Mostly, it was people who went to an event as an attendee, had a really great experience and wanted to bring it back with them, because people were traveling from all over. You had folks flying or driving from California, to Pennsylvania, to go to a hackathon. They were wanting to see more of it. Once a year is not necessarily enough for them to fuel that drive that they have.

[00:29:39] JM: Did it start out as a business? Did Major League Hacking start out as a business?

[00:29:44] JG: It started as a business out of necessity. We don't necessarily – I come from a place where we just have a bunch of money sitting around. In the early days, we just got money from our friends, to be totally honest. We had a lot of folks who are developer evangelist and we found ways to help them scale what they were doing with all of these hackathons. As it evolved, it became much more structured. Over time, we really solidified what our values were as a

company. We really solidified what our products were. That's allowed us to grow. It's always been a business.

We strongly believe that it is both important for our employees to have a meaningful stake in what they are creating, meaning equity. I really do think that's important in what people get out of being involved in early-stage company. I also think that's really important to be self-sustaining. Nonprofit is something that we floated many times. We talked to people about it. The biggest concern we had was you're dependent on grants and this general, societal trend of do people want to give money to this thing?

We felt if we were creating value for the participants and for the organizers and for the companies who are involved, which was part of the model before MLH, then we would be a more sustainable organization, because we're a business.

[00:31:21] JM: You said you eventually figured out what your products were and what your ethos as a company was. Tell me about the process of finding those things.

[00:31:35] JG: It was a lot of trial and error. In terms of the ethos as a company, we knew what that was. It was something that we had internalized long before we started MLH. It took quite a while for us to write it down in a way that was, I think digestible to people who weren't us. It's really hard as a founder to articulate your values in a way that other people relate to. We eventually did through, essentially consensus of our team. A lot of it was taking things that we did, or memes that were around the company and documenting what that actually meant to us.

We have five corporate values. I'm always a little bit of a cynic when it comes to corporate values, but these are things that very much represent what we were already doing that we just wrote down. We take out the trash, which is this idea that literally when you go to events, sometimes you have to take out the trash. It's your job to be helpful. It also means that we sometimes have to do the dirty work of running a company.

Sometimes our CEO is staying late and cleaning the office. Sometimes I'm here on a weekend and I'm doing something that someone else couldn't do, because maybe they had a family emergency, right? We help each other. We go the extra mile and we do what needs to get done.

Our other value is hackers first. That's really the most important thing that we have, because it's considered in all of our decisions. Everything we do has to take into account. Does this actually help hackers? Because they're our core constituency at the end of the day. We also have MLH provides, which to us, means that as a larger organization working with all of these different student groups, we typically have more resources available and we need to leverage those resources to help all of our local communities.

Because we might have a relationship with a company, or we might even have money that would make a huge difference in this local group's day-to-day lives. It's up to us to actually make use of that in a positive way.

Our next value is that learn, build, share, which to us means that we constantly should be stretching ourselves and learning, to be building new things all the time and we should be documenting that and sharing it an open way. It's really interesting. We have a open-source hackathon organizers guide that anyone in the world can either contribute to, or utilize. Doesn't cost anything. Doesn't have to actually be working with MLH. It's just a resource we felt should be out there.

Our last value is standard of excellence. This is actually something that I borrowed from previous jobs that I had, where your co-workers should be the first people to hold you accountable for doing good work. It's not your manager's job to be looking over your shoulder and saying like, "That's not good enough." Your co-workers should be the one saying like, "Oh, have you thought about it this way? Have you done something to the fullest extent that you can be expected to?"

Everything we do needs to be held to that high standard, whether it's a random e-mail com to a hacker who e-mailed us, or literally a huge corporate partnership. We need to hold it to a really, really high standard. Those five values are all things that we had done. They were frankly things that we would say to each other as memes like, "Hey, man. This has standard of excellence. Have you really gone through this with a fine-tooth comb?" When we wrote them down, it was pretty obvious at the end of the day.

It was super important for us as a business to articulate that, because over time, the people that you're hiring and the people that are joining your team are not necessarily from the community. They can't read your mind, and so you have to have a shared language for describing how the company works. We decided to put that into legal hard coding in our organization by becoming a B Corp. B Corps are a new model of company that allow you to be mission driven and for profit.

There's actually two ways to be a B Corp. The first is through a certification. That is extremely extensive. We had no idea when we got into it how extensive it was. They were like, look at everything from what's your parental leave policy, to do you recycle in your office, to what is the mission of your company and how do you fulfill that? It really picks apart every piece of your organization. Going through the process of becoming certified made us think through things that I think most companies just don't even know to think about.

Once you become a certified B Corp, you have a limited period of time to convert to a public benefit corporation, which is actually a legal designation in the Delaware Corporation Law that's different from a C Corp, or an S Corp, or an LLC. We are Major League Hacking PBC.

That means that if it ever comes down to it, your shareholders need to consider your mission, alongside the profit motive that all companies have to consider. That's super important, because it creates defensibility in our values over a long period of time, regardless of who's running the company 20 years from now.

Our mission, that's hard-coded into our bylaws is we empower hackers. Our mission is to empower hackers. That's how we measure the success of what we are doing and making money and the profit motive is our means to an end to accomplish that.

[00:37:16] JM: What has that meant in practice, when people have been empowered by your event and the work that you've done? What has that empowerment look like?

[00:37:31] JG: We look at it pretty broadly. Someone who goes to a hackathon and has a good experience, that is empowering someone. Someone who learns a new skill through a workshop that we've run, that is empowering someone. Someone who organizes an event, that's a huge

leadership opportunity. We are empowering hackers that way. The lens that we look at it through is A, does this pass the sniff test?

Does someone feel positively about this and get value from it as an individual? Then also B, is this a sustainable thing that we can do in the long term?

Sometimes it does mean that we're doing something unsustainable. There are times where I've been at a hackathon and someone comes up to me and they're like, “Hey, man. I'm having a lot of trouble. I can't afford my bus ticket home. I lost my wallet.” It's like, “Okay, great. I'll just expense you a bus ticket.” A lot of companies, that would be pretty difficult to do from an actual policy perspective. For us, it's something we do all the time. I think that's really, really important.

The other part of it is internal. It's how the team looks at itself and our day-to-day activities. As I said, as we've grown more and more people who work here don't come from the hacker community. They don't have that in their blood. They're often passionate about the mission of working with students and working in a organization that is mission-driven. What that means to different people could be different. Having those values and having the core mission is a guiding light for people. It's a framework for thinking about what they do.

For example, we have a team here called our ops team. They're doing everything from booking flights and hotels for our staff, to literally shipping laptops and Arduinos to events for people to use. Sometimes their thing that they have to accomplish is like, “We need to save a little money.” Well, when they're making that calculation of how do we actually save costs here and support more events, a big part of it that comes into play is okay, how do we save money while still providing a comparable hacker experience? Because it can't be one or the other.

The way we look at that is like, okay, let's say we have Arduinos at an event. How many Arduinos does the average event utilize? Oh, okay. They only utilize nine and we usually have 10 available. Great. If we cut it down, that won't impact the experience, right?

Those types of things really, really come into play. Whereas, I think a company that didn't have that as their guiding light could easily be like, “We don't need Arduinos anymore. They'll save us

a $100 an event." For us it's like, what is the minimum thing we can do that will still actually fulfill our mission, when we're talking about cutting back.

When we're expanding it's about, okay, how do we serve the most people possible? That's a global thing now. We have events going on literally on every continent. We have the one at a research base in Antarctica, right?

[00:40:44] JM: Seriously?

[00:40:44] JG: Yeah. We had a bunch of hackers who are at the McMurdo research base. Last December, they held a local hack day there, which is our single day multi-site event all over the world. On that day, we have an event going on on every continent, right?

[00:41:00] JM: What do you hack on in Antarctica?

[00:41:02] JG: Polar bears. I don't know. I mean, they're scientific researchers. They're working on a lot of it is climate related now. Some of it is experiments that can only be done in an environment that's literally that cold and dry. Some of it is I believe military, or defense related. The installation there is I think, has scientists from a lot of different countries. Not my main area of expertise, but they were primarily working on science related projects.

[00:41:35] JM: I'd really like to get an understanding of the business. You talked about some of the different products you have, like you have a workshop, you have actual hackathons, or is it a hackathon recipe, or you actually do the hackathon for people. Give me an understanding of what you sell and how the business has evolved over time.

[00:41:59] JG: The way MLH's business works is by partnering with local hacker communities. We have three core products. There is our hackathon league, which is what we've been doing for the longest and it's our biggest community. We have our MLH local host workshops, which are technical workshops in a box. You can run them at a meetup, a club meeting, 90 minutes-long, it's curriculum we've created. Then the last product is local hack day. That's what I mentioned earlier, where we have hundreds of sites on a single day doing the same type of event.

The way each of those works is through partnership with local hacker groups. Our services are free for those organizers and students to utilize. That was a very intentional thing, because honestly, college students don't have that much disposable income.

For a hackathon for example, they'll apply to work with MLH maybe six months before. We have a published list of criteria, qualifications, what the process looks like, what we get in exchange. We go through this whole process with them, where we qualify them, we publish them on our website, and then we're literally working with them as mentors leading up to their events.

Different events need different levels of help and mentorship. Sometimes it's a call in the middle of the night. It's like, “Hey, our biggest sponsor dropped out. Can you pull some strings?” Sometimes it's, “Hey, we're really having trouble hitting our budget,” right? We need to get buses, but the buses are too expensive. How do we deal with this situation? We have a team of people whose entire job is mentoring hackathon organizers, which is really cool that exists at all.

The way it's funded is each of the hackathons has local sponsors. Standard to any hackathon you've ever seen, they sell sponsorship slots to developer evangelist, recruiters, really whoever they want. One of the interesting things is we don't really get that involved in that process. We’ll intro them the sponsors, we’ll give them recommendations about how to structure their packages, but they own that budget. They own that revenue. They can spend it on what they decide. That creates differentiation in the events. Some events have t-shirts, some events have hats as swag. That's up to them.

[00:44:20] JM: It’s like a franchise model. The organizers get to make money off of their events?

[00:44:26] JG: I've never heard of an event where the organizers profit off of it, but they control their own budget. The thing to remember is that most of these funds are being processed through a school club bank account. It's not a for-profit thing that the organizers get. It's more that they control their own finances on a local level. That's a big part of the model.

Then MLH sells global, or national sponsorships or services on top of that to companies. For example, you're Microsoft, Google, Amazon, you're probably sponsoring a handful of local events and sending your developer evangelist to them. MLH provides a scale component that's difficult to achieve on your own.

For example, maybe every student who walks into a hackathon, regardless of where they are in the world, gets a cloud hosting credit, or a free domain name. That's something that we uniquely can provide that would be basically impossible to coordinate on a one-on-one basis with 250 different events. Then we standardize that. It's a very successful model. We've been able to drive a lot of really cool projects and interest from the student side.

It's interesting, because it's super mutually beneficial, right? You're a college student, we're basically saying, "Hey, you don't have to use it. We're not going to require you to do this, but here's a bunch of free credit if you want it one day," which is really fun. I would have loved to get a bunch of free credit when I was in college. It's really just a mutually beneficial benefit that we can provide because of the scale we have. At local host and local hack day, same exact model. Organizers, free for them. We have sponsorships on top of it that really utilize MLH's scale. They'll have local sponsors who pay the operating cost of the event.

[SPONSOR MESSAGE]

[00:46:22] JM: As a software engineer, chances are you've crossed paths with MongoDB at some point, whether you're building an app for millions of users, or just figuring out a side business.

As the most popular non-relational database MongoDB is intuitive and incredibly easy for development teams to use. Now with MongoDB Atlas, you can take advantage of MongoDB's flexible document data model as a fully automated cloud service. MongoDB Atlas handles all of the costly database operations and administration tasks that you'd rather not spend time on, like security and high availability and data recovery and monitoring and elastic scaling.

Try MongoDB Atlas today for free, by going to mongodb.com/se to learn more. Go to mongodb.com/se and you can learn more about MongoDB Atlas, as well as support Software Engineering Daily by checking out the new MongoDB Atlas serverless solution for MongoDB. That's mongodb.com/se. Thank you to MongoDB for being a sponsor.

[INTERVIEW CONTINUED]

[00:47:44] JM: This must have been an emergent business model. It sounds like you just jumped into the hackathon space, you had a sense for this being an important area to invest in, or figure out a business around, but that business model that you just described could not have existed from day one. Can you tell me about the process of – because this has been my experience with my software engineering podcast company, figuring out the business model was really an emergent process.

Can you just tell me about your personal experience in more from the building of business angle, how you explored the fissures within the world of hackathons as you were looking for a business model with enough potential to scale your company?

[00:48:35] JG: We typically do it through trial and error. What I mean by that is we will go to a company and say, “Hey, we have this idea that we think is going to be super beneficial for you.” We go through the whole process. We pitch it. We let them know this is the first time we're doing this. They buy it, we create it. If it works really well, we sell it to other people. In that sense, yeah, we're figuring it out as we go, like most startups.

What I will say is the types of sponsorships we sell now are actually surprisingly similar to what we sold day zero. What has really evolved is our understanding of the market and how you package this, how you productize this, what the ROI is for companies.

We frequently hear that we have much better ROI metrics than your typical sponsorship, because we are super data-driven internally. We really do care. How many students actually utilize the service? Interesting. Well, if not that many students utilized it, maybe they don't want it, right? There's certain things there that we think about in terms of future partners that really informs our decision-making.

What I'll say from the hackathon side, like funding the local event sponsorship, it is a direct correlation to the hiring environment in the world, in that developers are an extremely high demand and a lot of local sponsorships are coming from local recruiters. It's not necessarily the companies even think about.

[00:50:09] JM: Local recruiters.

[00:50:10] JG: Local recruiters, meaning your company has an office in Chicago, they sponsor all their surrounding hackathons. Not local recruiters, meaning like, contingent recruiting firms. That's not really a thing for students. It's interesting, because 10 years ago the environment probably would not have had enough demand to support that, but the demand is only increasing now. That is what allows a lot of these events in non-tech hub cities to exist and be successful, is that there is such a hiring demand that people need to be branching out beyond their traditional career fairs, or job boards.

[00:50:50] JM: Yeah. I imagine, every capital city in the United States has software companies that are really looking for good developers. At this point, there are probably enough software developers, young software developers in all of those capital cities that it would make sense for those companies to sponsor a large top of funnel recruiting engagement. Do you have any benchmarks, or something – I know coding and programming education is growing dramatically. I don't have a sense for how fast it's growing. Do you have any benchmarks, or numbers, or ways that you have – or just little signs of the times that you might have noticed recently that you think about when you're trying to get a sense for how rapidly the world of software is growing?

[00:51:46] JG: Yeah. There's a couple of stats that I always look at; one is the number of people getting CS degrees, which to be honest, only half of our audience are CS students. Not fully representative, but it's a good signal for the general trend. There's something on the order of 40,000 undergraduate CS degrees in the US awarded each year, and something close to 400,000 job openings, right? There's a huge disconnect there and the demand is really outpacing the supply.

I actually am on an advisory board for the CS department at the university I went to, which is Stony Brook. To be clear, I have a history degree. I ended up on the advisory board purely through the work I do with MLH. I was in one of the meetings recently and they were talking about how they don't even have enough professors to fulfill the demand for CS classes that the school is experiencing right now. I've heard that many other schools as well.

So many people want to learn this and many of them aren't studying CS, and so their schools are having to adapt really, really quickly. That's why you see such a proliferation of boot camps and extra-curricular learning opportunities is because, they can't fulfill the demand immediately.

The other thing that I look at is emulations numbers, right? This year 2019, we're expecting to have about 125,000 people come through our events. That's a crazy number. It is three times the number of undergrad CS students in the US. Where are all those people coming from, right? When we look at over the five years of our company, something 5% to 10% of all programmers in the US have been to an MLH event. You have this crazy explosion of interest in programming, CS, learning to code.

I think that it's all feeding into each other, right? Code.org doing hour of code at elementary schools, feeds into people wanting to do robotics competitions in middle school, which feeds into people wanting to do hackathons in high school and college. Over time, that has compounding value. When you start looking at emerging markets, like APAC, or African countries, it's only more so, right? Because programming is one of those things that is a huge way to get ahead from a life and earning potential standpoint.

More and more people all over the world are trying to learn programming, because it gives them opportunities. It really does open the door to new worlds, new opportunities, and especially with how easy it is these days to be an online virtual contractor for companies. It's not just contained to the Bay Area anymore. You could be a developer in Mexico City earning a San Francisco salary, because you're working for a major tech company in the US, or a major tech company in Mexico, right? It doesn't matter, because that's a worldwide trend.

[00:54:47] JM: How do you scale your company? We're sitting in your office right now, you've got probably what? 30 people working here. 50 people?

[00:54:56] JG: MLH is about 20 people full-time. We have another 50 or so contractors that we call coaches. They're typically junior developer evangelist, or college students who staff events for us, they produce content for us, they test different curriculum for us. They have a lot of different jobs. We look at it like a developer evangelist in training program. We pay them for it. Yeah.

[00:55:22] JM: Very interesting. Yeah. I mean, so I've tried to – Software Engineering Daily, like my – I want to make a bigger media company, but I've always had this feeling of a strain, because why would somebody – they could go get a job as a software developer and software developer jobs are probably going to pay better than a software media job, or a developer evangelist job, or a job at a hacking company, a hackathon company.

Is it hard to scale an organization like that, or are you able to find people who this job is so specific and it is – it just uniquely fits what they're looking for that doesn't really matter that they're not going to get the Google level salary and the Google food bar. Yeah, tell me about how you have scaled the recruiting process.

[00:56:18] JG: When we were starting MLH, I used to read all these articles that were like, "Recruiting is the hardest part of a founder's job." It honestly didn't sink in, until we actually had to do it. It's really difficult. As far as I understand, it's really difficult at a big company too. For us, we aren't just hiring developers. We are hiring people who have operations backgrounds, who have event planning backgrounds, who have community management backgrounds. When you're looking at the total available pool of people, it's actually much broader.

Some of the folks who work here used to work in refugee management jobs. Some of the folks here used to be teachers. Some of the folks here used to be event planners, wedding planners, that kind of thing. It does open up a lot of doors for us in terms of who we can hire, because we're not just trying to hire the same batch of programmers everyone else's. Frankly, we're not a VC fuel crazy growth company and we're not trying to be.

We are trying to be a sustainable long-term growing organization and that we don't necessarily want to like, I don't know. We don't really want to do that artificially. The people who work here

are often working here for many years. This is something that they like and they like the people they work with. They enjoy what they do and the community they get to see. That makes a huge difference, right?

It's one thing to go into a job and sit in a cubicle and write code every day for a product that you don't really care about, it's another thing to go somewhere where you get to go talk to students who love your company, right? That's a super different experience.

At least for me, even as one of the founders, that's one of the things that keeps me going is like, I go to a hackathon every once in a while and there are kids there who say like, “This changed my life.” Or maybe it's not even that serious, but they're like, “This is the best weekend I've had all year,” right? That makes a huge difference in your motivation and your enjoyment of your job, because you spend most of your time at work.

[00:58:26] JM: Yeah, it's interesting. There are all these coding-adjacent jobs, or programming-adjacent jobs. That's an emergent world, because as you've already said, the world of programmers is expanding so rapidly and there's all kinds of communications jobs and organizational jobs and operational roles that are adjacent to the vast scaling of the programming world.

As we begin to wrap up, I want to come back to education. I've done some shows recently about some newer education models. We've done a bunch of shows about boot camps. I did another one recently about a boot camp in Vietnam. Basically, the coding boot camp model that has worked in the United States applied to Vietnam and it's working really well. Then I did a show about lambda school recently. Given that you're on advisory board on the CS department, they must be aware that their curriculum is lagging behind, right? What's your interaction with them? Are they figuring out how to update the university curriculum?

[00:59:42] JG: I do think that they're trying to update the curriculum. Well, one of the things I actually learned recently, which I had not known about before is they have a class at Stony Brook that I don't remember the name of it, but I'll describe it and it'll make sense. You go to the class, and at the beginning of the semester they pair you and a couple teammates up with a local nonprofit. You just spend the entire semester building software for that nonprofit.

[01:00:06] JM: Wow.

[01:00:06] JG: I saw that and I was like, “This is amazing.”

[01:00:08] JM: Yeah, that’s perfect.

[01:00:09] JG: This is actually what we're trying to do at hackathons too, right? I think that you start seeing those seeds at a lot of universities. The other thing to remember is universities, especially public universities, are largely funded by research money. A lot of curriculum is designed to funnel people into research positions, more so than I would say, engineering tactical positions, right?

Success for a university doesn't necessarily mean that all of their students become software developers at a big company. Success often means that they are working on a research project that gets funding.

[01:00:50] JM: That’s a real conflict of interest.

[01:00:53] JG: I mean, maybe. I don't know if it's a conflict of interest.

[01:00:56] JM: I mean, don’t most computer science students, they view it as a trade school.

[01:01:01] JG: I think that's increasingly true. Yeah. I don't know if it's a conflict of interest, as much as it is parallel goals. The school wants research students. The school also wants students to get jobs. Sometimes those are conflicting. I mean, any good university, these days they're probably seriously thinking about what is our post-graduation employment rate, right? You talk about stuff, like student loans, you talk about stuff like all of these unemployed people who have college degrees and universities really do care about that. I don't know if they have direct specific solutions right now to fix it, but it's on their minds absolutely.

[01:01:42] JM: Last question, what's the biggest vision that Major League Hacking could evolve into?

[01:01:51] JG: We look at this as a global movement more than anything else. If we think 20 years down the line, I think that this is going to be the mechanism through which the majority of programmers in the world get exposed to the community, the fun side of programming. I think that we will be supporting crazy projects that people want to build, almost in a research and development capacity.

I think we'll probably be helping people get startups off the ground. I think that we'll be introducing people to programming as a means to an end for a wide variety of disciplines. Frankly, I think that we're already starting to see this, but I think that we will have a profound impact on CS education.

We're starting to see the early seeds of that where a professor will give you extra credit for going to a hackathon. I think in the future, it could be a core part of curriculum. Why shouldn't there be so many more courses like the one I mentioned at Stony Brook, where you just build a cool project and that's your learning for the semester.

I think we're going to be the driver for that. I said we're going to support 125,000 students this year. There's something 18 to 20 million developers in the world and more over a year. There's a pretty big pool of people that we want to impact. It'll take us time to get there.

We've been around for almost five years now. This trend is not going away, right? Software and technology are a part of every company's DNA. The ones that they're not part of yet, they're killing. It really is this fundamental shift in how the world works. It's one of the most valuable skills you can have. I think that we are a vehicle for people that enter that world in an ethical and positive way.

[01:03:51] JM: Cool. I'm a fan of what you're doing. The ways that I have learned programming the most, in a most condensed fashion, are when I have freedom to explore ideas and when I can socialize with other people about those ideas. The way that I learn is so far removed from sitting at a desk and taking instruction from people. I'm glad you're doing this and continue the success.

[01:04:19] JG: Thanks, man. I'm glad I got to do this every day too.

[01:04:21] JM: Cool. Awesome. Thanks, Jon.

[END OF INTERVIEW]

[01:04:27] JM: When I was in college, I was always looking for people to start side projects with. I couldn't find anybody, so I ended up working on projects by myself. Then when I started working in the software industry, I started to look for people who I could start a business with. Once again, I couldn't find anyone, so I started a business myself. That's the podcast you're listening to.

Since then, I've found people to work with; on my hobbies and in my businesses. Working with other people is much more rewarding than working alone. That's why I started FindCollabs. FindCollabs is a place to find collaborators and build projects. On findcollabs.com, you can create new projects, or join projects that are already going.

There are topic chat rooms where you can find people who are working in areas that you're curious about, like cryptocurrencies, or React, or Kubernetes, or vue.js, or whatever software topic you're curious about. We now have github integration, so it's easier than before to create a FindCollabs project for your existing github projects.

If you've always wanted to work on side projects, or you want to find collaborators for your side projects, check out FindCollabs. I'm on there every day and I'd love to see what you're building. I'd also love if you check out what I'm building. Maybe you'd be interested in working on it with me.

Thanks for listening and I hope you check out FindCollabs.

[END]