EPISODE 673

[INTRODUCTION]

[0:00:00.3] JM: The dev community is a platform where developers share ideas, programming advice and tools. Ben Halpern started it after running an extremely successful Twitter account, creating numerous humorous tweets for developers. He still runs that account. One way to describe dev community is as a cross between medium and Stack Overflow and Reddit, but it has its own personality. I recommend checking it out. It's dev.to. There's a link in this episode.

The dev community was open sourced and in today's episode, we talk about the challenges in the opportunities of having an open source social network, or platform, or tool set and we talked about his plans for the future, and where he is taking the dev community. Ben is an entrepreneur who tries a lot of different creative experiments and creative projects. His perspective has always resonated with me.

This is an example of a project that is really taking off. He's got a lot of traction, he's doing really well and Ben has been on the show a few times before. We've talked about the state of developer media and side projects and the identity of the software engineer. I think Ben and I met at similar times in our lives, about three years ago or so, where we were each having some level of success in our careers, but we wanted to start something.

I think we're both happy with how things have worked out, both of our experiments in developer outreach and business building have worked out. To some degree, I think Ben would agree with me, it was a matter of conviction and taking a plunge and being strategic about that plunge, so we've always gotten along well. Enjoy while we have him on the show. I hope you enjoy it.

[SPONSOR MESSAGE]

[0:02:13.2] JM: DigitalOcean is a reliable, easy-to-use cloud provider. I've used DigitalOcean for years, whenever I want to get an application off the ground quickly. I've always loved the focus on user experience, the great documentation and the simple user interface. More and more,

people are finding out about DigitalOcean and realizing that DigitalOcean is perfect for their application workloads.

This year, DigitalOcean is making that even easier with new node types. A $15 flexible droplet that can mix and match different configurations of CPU and RAM to get the perfect amount of resources for your application. There are also CPU-optimized droplets perfect for highly active frontend servers, or CICD workloads.

Running on the cloud can get expensive, which is why DigitalOcean makes it easy to choose the right size instance. The prices on standard instances have gone down too. You can check out all their new deals by going to do.co/sedaily. As a bonus to our listeners, you will get a $100 in credit to use over 60 days. That's a lot of money to experiment with.

You can make a $100 go pretty far on DigitalOcean. You can use the credit for hosting, or infrastructure and that includes load balancers, object storage, DigitalOcean spaces is a great new product that provides object storage, and of course computation. Get your free $100 credit at do.co/sedaily.

Thanks to DigitalOcean for being a sponsor. The co-founder of DigitalOcean Moisey Uretsky was one of the first people I interviewed and his interview was really inspirational for me, so I've always thought of DigitalOcean as a pretty inspirational company. Thank you, DigitalOcean.

[INTERVIEW]

[0:04:20.1] JM: Ben Halpern, you are the founder of Practical Dev. Welcome to Software Engineering Daily.

[0:04:24.6] BH: Yeah, thanks for having me Jeff.

[0:04:26.3] JM: We've had you on a couple times in the past. We met about three years ago. You had this Twitter account with a lot of traction and you and I were both thinking about media for developers; social media, news media, essentially the types of content and the types of

conversations that developers wanted to consume. What was your outlook on developer media back then, about three years ago when you started tweeting a lot?

[0:04:59.0] BH: Yeah. My general outlook just has definitely evolved a little bit since then. I think at the time, I was in exploratory mode. I felt the landscape was best confusing, unhelpful, or just generally could be improved. I have a hard time even looking back on that time with that totally painting a bit of a narrative fallacy on the whole thing. At the time, I was really just a blank slate. I was really thinking more about the software industry itself in a way then how everything really fit together, but I also separately knew that I had a pretty good handle on the media landscape and just how people are interacting and maybe where things were going. It was a little bit of everything. It's hard to really put myself in that exact mind frame anymore, because it's just a lot has happened since then.

[0:05:52.1] JM: Okay, so in the present day as there was back then, there are the distribution channels and then there are the actual source channel. The distribution channels like Reddit, Hacker News and then the actual source channels like Quora, Medium, or New York Times, Washington Post, these are all different forms of media that developers consume. There's also Stack Overflow of course, that's more like a reference manual. What were the gaps that you were seeing back then and what are the gaps that you continue to see today?

[0:06:28.0] BH: Yeah, absolutely. This is all framed the way you're putting it in terms of your universe, which is perfectly fine. It's definitely, I feel you can attack this from different angles. In terms of – there's the input, there's how I want to express myself as a developer, there's the tools I might use to do that, there's the different components in terms of where the audience is, how much do I actually care about the audience compared to just am I doing this just for my own self? What does the software development industry need to work?

Media in our industry is a lot different than other industries, because it's really – it's marching orders a little bit. You hear about what's going on in a certain community, or a certain technology and you know what you are supposed to do with your life. The landscape has always been a little bit all over the place, because software developers are a little herding cats, they're very opinionated in certain ways. There's an xkcd about even in after the singularity happens in

technology, someone's still going to want to connect via IRC, instead of whatever the other platform is.

There's just a very human element of chaos in software development. We also are early adopters of a lot of different platforms, and the whole thing winds up being a little bit of a mishmash and that's definitely still the case. I thought that in some capacity, we were all using platforms that we're not really built for our needs and really designed around our particular use cases, but they also – we don't want to use these hyper focused dashboards and stuff. We want to use the same thing other people are using, in terms of just simple inputs, editors, social media patterns and stuff like that.

On that front, that was definitely where my head was at. I felt frankly, it wasn't the most groundbreaking ideas, but I was the right person to do it and I think that's what really got me excited. I felt I had a good grip on the whole problem. From the – an idea of the landscape, right down to the capacity to code it up real good. At the time, my ambitions were I think less. It was a side project. I was really just trying to scope out the situation, spend my time in a way that was interesting and fun, then also just throw myself out there into the software industry.

I think before I even started this project, I was basically totally anonymous in the industry. I had about a 100 Twitter followers, I didn't necessarily put myself out there in any way in software development. I just followed the trends, but I kept growing more and more towards the idea that I did have a lot to offer. I had a perspective, a background that could really help things, and I wanted to reflect that in not just a personal blog, but a real project that was devoted to this space.

[0:09:23.2] JM: One thing you and I have talked about in the past is the problem with narratives around what a developer is, what a developer is supposed to be. Historically, these narratives were shaped by corporations, or by university computer science programs, or more recently, by Y Combinator, or Product Hunt. In actuality, there's a much wider spread of what a developer can be. Does the practical dev represent a different narrative around what a developer should be, than some of these pre-existing narrative structures?

[0:10:07.3] BH: Yeah. Before I specifically get to that, I want to emphasize that we typically have dropped practical from what we talk about, except the Twitter account and our social presences and stuff. I feel this is perfectly good material for the show just to talk about this.

[0:10:25.0] JM: That's good, because I am an impractical dev, so now I am more welcome in this community.

[0:10:30.3] BH: I really did start the whole thing, just I wanted to get some practical ideas out there. The practical dev seemed like a good name for the whole thing, and still what we call ourselves, but it evolved really naturally from there. I developed this logo that was just DEV in all uppercase, so that's how we talk about it now, is DEV all uppercase. Then we also – everything is really centered around the platform, which we called DEVto, which is dev.to and it's pronounced dev two.

For a long time, I just let people talk about us however they wanted to, because it's their thing, not mine how they describe us. It's really not really me to say, but recently, I've just started correcting people and just saying call it dev.to if you're talking about the platform, or DEV if you're talking about the whole thing.

Our official incorporation is DEV Community Inc. That's where we are. We're a DEV Community. That's an interesting point depending on where people got looped in on the whole thing, they naturally just say dev, or practical dev. It's all good. I just felt I've been on a mission during these podcast appearances to really give the canonical name. We can say dev.to, or DEV. Anyway, what was the question? Sorry.

[0:11:41.2] JM: No, absolutely. I'll acknowledge your rebrand from here on out. The question was around the existing narratives around what a developer should be and whether DEV Community represents a different narrative around what a developer can be, or what a developer should be.

[0:12:01.1] BH: Yeah, absolutely. There's always a ton of pressure in every space to really fall in-line, or follow the leader and stuff like that. In this space, that's a really a fool's errand, because every different situation is different. There's so much you can do with software. You

can really fit it to whatever your lifestyle is, and maybe that's not an opportunity to retire to the beach for everybody, if that's the lifestyle you want. If you're interested in being exploratory, working hard, there's really every possible space.

There's a place for everyone in this industry and a lot of the narratives are written by the most famous popular developers, or by #thoughtleaders who don't necessarily live the life every day. There's absolutely a ton of room for individuality for just everybody really being themselves. The whole industry is so much more mainstream now. Jeff Atwood came on your show forever ago and I remember he said, "The geeks won in a sense. Software became a the biggest thing in the world." Then it also got distributed, like everyone can do it. Not quite there yet, in terms of equitability in the industry, but that's where we're going and my entire part of the industry is trying to help things actually reflect the mainstream element.

Things aren't done the same way they were done in the 90s, or the 2000s, or the 80s, or any time before. It's always evolving and things are right out there in the open, like my parents' friends might ask me about some particular software thing they probably just heard on the news, but it's happening. This is all pretty mainstream and our job is a pretty normal job that every company has at least, if they don't employ software developers, they work pretty closely with them, if they're of any size.

Software is obviously a huge deal and yeah, there really aren't any solid in single narratives that if they ever existed, we're trending towards a much more just, like software is just a part of people's lives and a part of a lot of different types of people. They're definitely no single type, or narrative, or path. You get benefits from coming from all sorts of different directions and I myself have a long and tumultuous relationship with software and technology. I've been interested my whole life, but I was in and out of how much I was interested. I took some CS in college and then I dropped out, and then I took the degree I had and went to get a job and I really hated it, so I got back into software. That finally stuck, because I got really into the certain type of stuff I wanted to do.

Yeah. That's the freedom and experience that I really want to try to push forward, and this project dev.to is all about how I felt I could contribute with my software skills, as well as the other

things I worked on, and the other skills I built up over time. Absolutely, it's a really good way to describe my purpose for some of the stuff.

[0:15:07.1] JM: Your process of getting started with dev.to was not completely straightforward. It was a matter of you running a number of different businesses. I think just two business, or one business and then you started dev.to, and then we're going back and forth between these two businesses, and then ultimately, your hunger to run dev.to won out and you transitioned full-time to it. What was the process of making that jump, in shuttering the other – and I think the other business had some traction, significant traction and you had to give up this thing that had some traction to go all-in on dev.to. Tell me about that internal debate and the process of making dev.to a full-time thing for you.

[0:15:58.8] BH: Yeah. A lot of it had to do with the various people in my life, I was committed to one way or another. I have good professional and personal relationships with everyone involved in all that stuff at the time. There were a few things that made transitioning to dev.to the obvious choice in the end. As much as everything else was going fine and in some ways dev.to is such a typical business. The other stuff we were doing was like, SaaS, it was really easy to explain to people. In some universe, it was very straightforward. Nobody questioned what I was doing with my life if I was working on more traditional stuff, like where software is just a very obvious part of the ecosystem and my software development.

A couple things, so I definitely became over time more and more impassioned by this whole thing. This is really how I identified to people. I would often go to conferences and tell them they knew me from the practical dev more so than anything else, but I had to tell them like, well on the badge it says I'm doing this other thing and I was very involved and I'm still working with those people. What we had wound up doing was we just switched. We merged the companies. I gave up a lot of my equity to work with the people I was already involved with, and it all was awesome.

Yeah, at the time it was very – I didn't really know what the outcome would be. I remember reading a blog post by Saron Yitbarek from CodeNewbie about how she went full-time on it. Not really involved me a little bit. I wasn't sure what she was doing with her software development

community, but knowing that she left Microsoft to do that full-time and she lived in New York at the time, so I got a cup of coffee with her and talked about it.

That really gave me – at that point, I wasn't even sure I was considering it a true business. I don't think I'd even – I knew I was keeping my options open, but that was really early and that really helped me. Then I met one of my two co-founders, Jess Lee that was the first person to be helping with the project. She brought a ton of discipline and a ton of skills that were not necessarily my forte, which really helped this get beyond the project stage. She just brought a lot of skills.

Then we were still – this was a side project, but when we launched on Product Hunt we were number one. It's just internet points being number one on Product Hunt, but you can't do any better than number one. All these are all signals back to us that this shouldn't matter, and that's when Peter, my partner in the other business came in got involved and we just had – we realized in every way that the world was captivated in some way by what we were doing and we had to give it everything we had.

It was much more really doing better things with DEV than we ever could have been doing with our SaaS business. Based on my own personality and where I come from, my family, I actually personally had a harder time justifying to people why I was working on the SaaS business, because like my mom said, “That's so boring. I can't believe you're spending your time on this.” I agreed with her in some capacity.

That's funny. A lot of people have the reverse influence from their parents, but I just have my own background and stuff like that. I'm the black sheep of the family in that I have a desk job of some kind. Everybody else is doing weirdo crazy stuff. Except my brother who's been doing music his whole life and now just recently got into software, so I pushed him that way. Anyway, so it's all just a personal journey. It takes you where you need to go and last, when we started this as a full-time, real business and we've been growing crazy ever since.

[SPONSOR MESSAGE]

[0:19:48.3] JM: When a bug occurs on your website, LogRocket captures the user behavior and allows you to do an instant replay to see how the user responded to the bug. LogRocket lets you replay what users do on your site, helping to reproduce bugs and fix issues faster. See issues as if they happen in your own browser with a full video replay and get those issues fixed fast to maintain the health of your application and keep customers happy.

LogRocket works regardless of your applications language, or framework and it provides SDKs for specific technologies. You can easily integrate with the tools that you already use. You can check out a demo at logrocket.com/sedaily, which would also support Software Engineering Daily.

LogRocket also records console logs, JavaScript errors, stack traces, network requests and responses with headers and bodies, browser metadata and custom logs. If you're triaging back-end errors, it can be unclear why the front-end made an unexpected request. LogRocket integrates with back-end logging and error reporting tools to show you the corresponding frontend session logs for every back-end error and log entry. Quickly understand your bugs and fix them with LogRocket.

Go to logrocket.com/sedaily to find out more to see a demo and to support Software Engineering Daily. Thank you to LogRocket for being a sponsor.

[INTERVIEW CONTINUED]

[0:21:31.7] JM: The thing about these community businesses is the margins on advertising revenue is incredible. You look at Facebook for example as the extreme example, and Facebook puts pixels up on a screen and charges advertisers for putting placements on those pixels. That's a really good business model, because it doesn't really cost you much to display pixels on a screen, and you get paid dollars for them.

On the other hand, you have to reach a certain volume in order to be able to deliver advertising that does sell well, and also there's a big question of whether advertising is a bubble and whether Google and Facebook will just capture all of the advertising, but there's also the other side of things where subscription-based businesses are getting much more viable on the

internet. People are more and more willing to pay for stuff, maybe they're willing to pay for a community, maybe they're willing to pay for other value-add services on a community, maybe some premium content or premium features. How are you thinking about the business model of DEV?

[0:22:39.4] BH: Yeah. To this point, we've really sustained ourselves with a couple different things. I think we're always going to be a couple of different things-driven business, because ultimately we're a community platform. It's not always so straightforward. To this point, we've really sustained on a few big sponsors and we really do differentiate between a sponsorship model versus an advertisement model.

We give a few big presents announcements for our sponsors as they get on board, and then we let their presence be known as a contributing sustainable partner for the community. Otherwise, we don't do a ton of display advertising, or anything like that. Then we do have what we call a sustainable membership with – a sustaining membership, which some people have gotten involved with which is just say a way to get a few key features, but not otherwise not dumb. It's not a necessarily driven by the features, but more so just a willingness to help support the project and really just get closer to the founders, like take part in a few smaller groups.

Ultimately, we continue to evolve what we're doing. Our vision for the future is really driven around really additive services for a software development career, and that's really the way I've started thinking about this in the long-term is that we're a career services organizations. Everything we do in terms of people's presences on the platform, the profile they're building, the network they're creating within our community, within our platform, within our network is really they're building their interests and their hobbies, but they're also building their careers with us a little bit.

We're working on things that really aid that, in the education space and also the placement space and everything. This is all very much just an ongoing effort to really do things nice and in a way that really respects everybody's interests and where they want to be spending their time. When I talk about placement and careers and stuff like that, the last thing we want to do is just let recruiters on the platform to annoy people.

We've been building in some features where people can opt in to just working with us for placement when they're ready and then opt out when they don't feel it. Some other little things and that's all really the most exciting part about what we're doing from a business model perspective. It's all fairly experimental. That's how we work. We don't really try to map the whole future app before it happens, but we've been creating some good partnerships and really doing this in a way that people have been – just every user that's gotten to work with us in this capacity has been just awesomely successful and happy about the whole thing.

When everyone involved is so excited to be involved, that's when we know we have a winner. In terms of the closest thing we've gotten to that point in the sponsorship area has been a contest, so we've done one official contest and we're working on a few more, but they've really been really fun. People have been absolutely totally excited to get involved and the organizations we've worked with have been over the moon about how much mind share they created by taking part in one of these contests.

A few things are working. We really lean into what the most smiles on people's faces and stuff and we've discovered really what works for us. The last thing we wanted to do when we embark on some sponsorships and some other revenue generating stuff was to lean into it, to feel that's the part of the business we had to optimize for. We took a good healthy approach to that, but then we tried to look elsewhere to expand on the business model. That's where we are today and that's where the most excitement lies in terms of our future is that we didn't let ourselves get too tunnel-visioned on the easiest way to make money right away, but we've been exploring and it's been incredibly awesome.

I just happened to be talking to founder of another more well-known company than ours and they just happen to mention that they found a great employee on our site, just through the natural stuff we do, in terms of where people can indicate they're looking for work, or little things like that. I didn't even know that happened. It just happened within the organic minimal stuff we've always had as features.

That's great to hear and we don't even always have a lot of vision into what's going on in these capacities, but we often get these awesome stories. It's just been magical. That's why I have a hard time even looking back to the beginning, because I truly feel like we've just been observing

our own community and really being helpful where we could and then letting them tell us what the world was like, especially since we're really a global community where it's – we'll often get stories about my situation as a developer in Nigeria is just a little different, but I have a lot to offer in these specific capacities.

It's just been just a pleasure to be a part of this whole thing. I can't believe how much more fun I'm having with this than the other things, which were perfectly good businesses, perfectly challenging engineering opportunities. I'd like all the people involved, but this has been so much more fun.

[0:28:01.5] JM: Well, there's an anecdote I remember when the Reid Hoffman was raising money for LinkedIn and he would pitch the company to venture capitalists. When he was pitching the business model, he would lay out four or five different revenue channels. None of which were working on the day of the fundraising, but if you fast forward to today, they're all working. You look at LinkedIn and it's bizarre, because whereas a company Google, or Facebook makes almost all of their money through one channel, at one specific advertising channel, well, Google advertising is broken up into several different components, but it's like 70% or 75% of it something is the search. I think in AdWords.

LinkedIn if you look at the pie chart of revenue, it's like four significant chunks; it's recruiting, it's some subscription sales services, it's advertising, it's premium subscriptions. LinkedIn has all of these different channels that they make money on. You can imagine something similar happening with DEV. If you manage to get the snowball of developers, of developers communicating with each other, expressing their interests, which are highly correlated with professional interests and professional opportunities and maybe even if it's not professional opportunities, its capacity to buy high-margin developer services, like cloud hosting, or SaaS, or stripe for payments or something like that. There's so many different services you can sell to developers.

There's a lot of different things you can do if you reach that critical mass of developers. I think the other interesting side of it is I've heard you talk about this a little bit, you can also just build your own tools to offer to developers. There's no way that Github and Stack Overflow are the endpoint of developer productivity across the internet. I don't know what the next manifestation

is, but it seems like every year, there's some new tool that comes out that's like, “Oh, I want this in my workflow, whether it's Slack or Asana, or I don't know, Air Table looks interesting.”

You could definitely imagine networked productivity tools emerging from practical DEV, especially given the open source virtue of it. Do you have any ideas for what form those productivity tools might take?

[0:30:40.1] BH: Yeah. That's definitely really in-line with – it’s I think a very obvious path for some of this stuff, and especially going back to what you said about the different revenue streams and then that's definitely how we see it. We try to think about things in at least conceptually, like bigger buckets, so that even if one element has a subscription thing and then also some other component, and we try to at least conceptually bucket things, so that it's one idea for the user and the member, so that's why career services is a bucket which encompasses a couple ideas in terms of training and placement and stuff like that.

Then yeah. I have been thinking a lot about the tooling stuff and where networks and productivity and these things all spin together. It's something I've purposefully not gotten too far down the rabbit hole, because we just have enough on our plates right now. I don't want to really give any of these any more weight than the others, but our platform really is a screen you're going to be using, so any monitoring, any alerting systems, we have a more custom holistic platform, then a lot of general purpose tools like Slack, or things like that. The more we can really build just the situation a software developer might need; so people come for the networking and the greater opportunities, but there's also a lot of their own workflow, which we expect could fit nicely in terms of logging and monitoring.

Also just implementing SaaS services in a unified authentication concept. There's some companies that do this already and there's certainly a lot of platforms that do this. We were hosted on Heroku and if I want to add additional services, they have that built-in in the same way that we could eventually potentially be the center of the universe in some capacity, where you can more easily manage your different authentications, your identities and different services.

These are all really interesting cases and I really get excited certain Sunday afternoons, really walking around the neighborhood just thinking about these things. It's a thing, like I'll maintain a certain worldview, but I don't want to really start building towards this stuff right away. We've been pulled towards building certain things at certain points, so especially given that we're an open-source project, we've had more and more conversations with organizations to integrate a little bit more. When I talk about monitoring and stuff, we're not going to start building every single one of these services ourselves. We're not going to build our own logging tools, or stuff like that today.

I mean, who knows what the future will hold, but we can actually integrate at a much deeper level with certain functionality, because the fact that we just have our code out there. I've listened to actually a few podcasts about how Linux really operates and how individuals who want to integrate with Linux, like if they want a specific just support system, integration, really any functionality that fits their needs, the Linux folks will talk to a lot of other people. They'll triangulate on the proper path that's going to be best for everybody and then they let people build these things.

We've already had that a little bit with our project, so we have a certain technology we use for search, but we could use other ones and there's a lot of pros and cons there and as soon as we open source, like I got a DM from someone who just worked at Elasticsearch and we just talked back and forth about and if they ever wanted to specifically really lean into helping us build the great search engine. That would be something that would automatically make us use their services as a company. If that works for the community and everything, we would definitely go that route.

We've talked to some other folks who might want to do business integrations. Some of them, if we agree with the idea, but it's hard to justify our own time to build it, we say like, “Hey, you can build this yourselves.” I mean, just the other day I got a DM from a business customer in a sense, like he's a developer relations person at a major developer tooling company, and there's been a lot of ways we've wanted to support them for a while, but we keep not getting around to building the features they really want in terms of just helping them manage their presence on the platform and stuff like that.

He just messaged me the other day, like right on platform, he messaged me through our messaging and he didn't ask whether this feature made sense, because we've already had that conversation several times. He asked me whether this database table makes sense for the PR he's working on. He just got around to starting to work out in himself, because he really wants it and we want it too, but we haven't – we have enough going on. In a selfish way, he's building something the rest of the community will really make use of.

That's so much of the magic of open source and that's really what took me from being like, “Oh, open source is great. It's cool. It's wonderful for the world in some ways,” but to the point where I was so convinced if it's power and utility for us in terms of building really something much more awesome than anything we could possibly build ourselves.

[0:35:56.5] JM: Well, the model of sponsored features, like Elasticsearch gets to build the search engine for practical dev. Then maybe they get some messaging kickbacks somehow, or you can imagine some integrated dashboarding tool that would want to build some internal dashboard for users of practical dev. Then it says, this dashboard is made by Mixpanel, or something. That's interesting. That's whether or not they pay you, maybe they just build on practical dev as almost as a way of building their billboard within practical dev. That could be an interesting way of accelerating the development and be some corporate stuff, but it works.

[0:36:47.4] BH: Yeah. Well, because our whole platform is open source, it's not even that we need to put their logo on things, although we'd be willing to do that just to let people know. It's actually helpful as a user to know, “Oh, this feature was built with this. That's useful.” Not that we wouldn't be open to that. It’s actually a pretty useful way to generate money and also help people understand just where things are coming from.

We are our open source, so we have a lot of people looking at our code and a lot of people contributing and they have to use some of these services in order to make their jobs work. Just by being the service provider for us, you get a lot of implicit exposure. Ultimately, we have a very good thing going and the last thing we want to do is really do anything that's only for us and not for the community. The outcomes here are really built around just natural good relationships, so this this individual building basically just redoing the RSS feed a little bit so that they can post

more naturally with their flow, in a way that everybody's been asking for, but we haven't had the space to do it.

This is just helping them be better users and helping other people be better users by proxy and it really works for everybody. Developer advocates, like nobody's time is so specifically devoted. You can you can say that software development time is very expensive, but the way people spend their time is very wishy-washy and abstract and you can't really nail these things down super well. This individual who's having fun building this features is doing this in a way that we might have – in another sense, we'd have to pay somebody a lot of money to help build this, but this just really works with their flow and they're building it because they want to be first to be able to use this feature and that stuff.

It's all very natural and there's a lot of behavioral economics under the hood. The big thing is that it's been a really healthy part of the ecosystem and we just consistently want to try to be additive and not take anything out. If there's ever any question about that, as far as I can tell, always maintain that perspective on everything we're doing.

It’s really been great. I just went to Medium today and I saw a particular paywall on a post which I'm pretty sure that individual is not being paid for – Medium is just and some of these other platforms, the Facebook's always been like this in terms of building out a feature that people get heavily involved with and then just completely pulling the rug out from under people. We've just completely been loathed to go down some of these paths, where eventually this was obviously going to not work out for everybody involved.

We've been a little bit more conservative about just our path and we haven't made as much. I mean, I don't think we've made a tenth of how much money we could have made on sponsorships, because that's not how we wanted to do business in five or 10 years exclusively, like it's a part of how we do things. We know that we're onto something really fantastic, as long as we can just keep a devotion to the users.

Part of the sponsorships has been maintaining our optionality and ability to grow the business how we want to. If we weren't making any money and we were just casting that aside and being like, “Oh, the money will be there, eyeballs, eyeballs, eyeballs.” That's the company that ends

up having to take absurd amounts of venture capital and give up board seats and stuff like that. We're keeping all our options available, and I think we will raise some money at some point, but the wonderful thing is that we're doing this all in a way where we really are able to build the thing we want to build, and we are able to give the users the experience they expect and do that and hold the cards and keep our board seats.

It's a three-person board; me, Jess and Peter, and that's going to be the case for a while and eventually we might bring out a partner who will want a board seat and we've had lots of people ask us for board seats just for advisory reasons, or whatever. We really regard the fundamentals of the business pretty tightly. It's been pretty great. I'm so glad I have founders like Jess and Peter who get on board with this stuff, because a lot of this stuff is just me imagining a future I want to see and they have to ask me some tough questions and then ultimately get on board.

Because the way we did things with open source and the whole thing was never this long conversation where like, we all came to it collectively. It was just that I had this ultimate idea as things change and as other opportunities come up. I have to go to them and make sure it works and they can obviously come to me if they have another vision for any future. My favorite thing is that I think we've built a really healthy company and a healthy community. I never feel like we have to do something for us that isn't good for them and vice versa. That's a bit – I have gone quite a few tangents.

[SPONSOR MESSAGE]

[0:42:09.9] JM: Kubernetes can be difficult. Container networking, storage, disaster recovery, these are issues that you would rather not have to figure out alone. Mesosphere's Kubernetes as a service provides single click Kubernetes deployment with simple management, security features and high availability, to make your Kubernetes deployments easy.

You can find out more about Mesosphere's Kubernetes as a service by going to softwareengineeringdaily.com/mesosphere. Mesosphere's Kubernetes as a service heals itself when it detects a problem with the state of the cluster, so you don't have to worry about your cluster going down. They make it easy to install monitoring and logging and other tooling alongside your Kubernetes cluster.

With one-click install, there's additional tooling like Prometheus, Linkerd, Jenkins and any of the services in the service catalog. Mesosphere is built to make multi-cloud, hybrid cloud and edge computing easier. To find out how Meossphere's Kubernetes as a service can help you easily deploy Kubernetes, you can check out softwareengineeringdaily.com/mesosphere, and it would support Software Engineering Daily as well.

One reason I am a big fan of Mesosphere is that one of the founders Ben Hindman is one of the first people I interviewed about software engineering back when I was a host on Software Engineering Radio. He was so good generous with his explanations of various distributed systems concepts. This was back four, or five years ago when some of the applied distributed systems material was a little more scant in the marketplace, was harder to find information about distributed systems in production, and he was one of the people that was evangelizing and talking about it and obviously building it in Apache Mesos.

I'm really happy to have Mesosphere as a sponsor. If you want to check out Mesosphere and support Software Engineering Daily, go to softwareengineeringdaily.com/mesosphere.

[INTERVIEW CONTINUED]

[0:44:29.2] JM: The open source aspect of it is really interesting, because a lot of these websites – the thing is, I don't know ad tech well enough to know, or I guess I don't know the social web sites well enough to know how important it is for their quality of product to be able to track you through the internet. Because that's one thing you're not going to be able to do with practical dev since it's open source is the people who are watching the codebase are not going to like if you insert some piece of code that tracks them throughout the internet and maybe that code is key to providing a high-quality targeted advertisements, but maybe you don't need targeted advertisements, maybe you don't need to track people throughout the internet, maybe that's not as important as people believe, or maybe you're you recover from it by having this degree of transparency.

Anyway, that's one microcosmic open source question. When I had Henry Zu on the show, you introduced me to Henry and that was actually a really good interview. We talked about open

source and how open source can be beautiful, but also painful. Henry goes through this with Babel and particularly working with the Github open source contribution tools, the open source moderation tools as well. Talking to Henry really made me think about the fact that Github, I love Github. It's fantastic, but probably there are going to be some other code collaboration tools, or program, or collaboration tools in the future.

What are the challenges that you've encountered so far of making a large open-source project work? I realize it's in the early days, but maybe you're already getting some hints of the types of governance issues that are going to be tricky.

[0:46:15.2] BH: Yeah. I think so far, we've been so wear our values on our sleeves, that the governance, or just in terms of some of the management issues have been great. We never expect to rest on our laurels in this capacity, but we've taken a very exploratory view on this. We haven't assumed to know how the future is going to work out. So far, I think just our general approach, like the way we've been managing this community the whole time really lends itself over to the open source project.

People have an expectation about how they should – we have an expectation about how people should behave themselves. Nobody is so important to the project that they get to be an – and maintain a position with us in any capacity. We've really gotten, I think ahead of this stuff. On dev.to itself, we have I think very sophisticated moderation tools. Even if they're just simple little like – it's a bit of a combination of crowdsource and then top-down admin stuff, and we have a few ways to deal with gray areas and get ahead of certain things.

The fact that this has been a core competency for us, I think we're just gotten pretty good at some of these things even if it's hard to describe exactly what it is. Then ultimately, things have gone pretty well. Just the other day, I clicked out to – someone referenced a different issue in a different project in our project and I went to another tab, got distracted, came back to it and then I read some of the comments and I forgot that I was on in a different repo. I read the comments and I was like, “Holy shit. What the hell just happened? Why is everyone yelling at each other for no reason? Why are people giving each other shit for simple –”

Basically someone included a code snippet and they didn't include all the right stuff. Instead of just giving them pleasant instruction about how they could be, like this issue could be done more helpfully. They immediately just attack them. I just was blown away. I was so shocked by this scenario. I knew this is how things are. The shocking part was that I thought it was in our repo and I was like, “Oh, my God. I got to do something right away about this.”

Then I realized I was in someone else's repo and I was like, “Damn. I'm so glad that this is not how we’re letting things happen.” We get to we get to think harder about this stuff, because we are a business and I get to put all my time and energy into making some of these things work. Whereas, a lot of open source projects, like you just don't have the time and energy to make some of these things happen and you're just doing the best you can.

Henry's full-time, but he's full-time on something that's infinitely more massive than him and he also has to spend his time just trying to make the money he needs to be full-time and it's really tough. I think the open source ecosystem, there's a lot of different things going on. Several components are tremendously sustainable and some are incredibly unsustainable. The whole space is just there's a lot going on and I'm just glad that we're more in this space and not just observers, so we can really get to solving some of these problems.

We are a company that tries to solve developer problems through social dynamics, through education, through helping people navigate the situations they find themselves in. I'm very happy that we're in this space full steam now and we get to really think hard about what the future is going to hold.

[0:49:43.3] JM: Such a little bit on the engineering of the project. What are some mistakes that you made in the architecture early on and what are some unusual decisions that have turned out to be rewarding? What are the pros and cons of the architecture you chose for practical dev?

[0:50:01.8] BH: Yeah. I can't speak to anything that seems like a big mistake, but I could definitely speak to things that are worse now because of the path we've traveled. Early on, I was very consumed with performance and remain consumed with performance. The client, I went with totally vanilla because I just wanted to experiment with the concept of no dependencies and

stuff like that. It was great and it worked for a while, but now we have to really nail down that that part of things.

Some of the code in that area of the codebase is just crappy, because it's vanilla, no frameworks, you're tackling every problem in a totally different way. It's totally a bit – some parts of that it are totally mangled, even if the product itself benefited from some of these decisions. We're in the process now working with people to improve things. I myself, I am not a strongly devoted front-end engineer.

I know enough to know the different bottlenecks and the important decisions and how this all fits together, but in terms of front-end development patterns, I know how to use the libraries we make use of and I know how to write good JavaScript, but I don't really pay attention to all the nuances there. I'm really being guided by the community and really trying to make some good choices there. As we grow, I hope maybe we might be able to bring one or two people on full time that truly does have a better grip on this stuff than me.

Overall, you just can't know everything, so there's a few warts, we're just like, “Shit.” I do not have any idea what the best practices are here. I'm doing my best and now we're really figuring that all out now. A few things are iffy, but I'm pretty happy with everything. We're a rails project, which is a weird choice for some people, because it's not the newest thing, or the most exciting thing, but I really thought it was a good choice. It's really easy to get involved with if you are in the rails community. There's some onboarding things we need to work on in terms of our own ecosystem, so it's easier for people to contribute, but overall, I think the long-term I think things are looking pretty good there, even though it's not the most forward-thinking stack you ever heard of.

I'm extremely excited by some of the other organizations that are now on the latest version of rails and are actively contributing back to the rails framework. The big one is Github themselves, who for a while has been working on a fork of rails and just hasn't been able to be as actively involved in pushing the framework forward. Then Shopify, I think as an organization has been the biggest contributor to rails lately. Github is obviously such a big player in open source, and the fact that they're now at 5.2 and actively looking to contribute back and they have Eileen

Uchitelle is the Github person who's also just central to everything going on with rails, and I'm just so excited by their involvement and everything now.

I couldn't be happier with the choice at this point. I didn't even know this would all come together so nicely. The future is just magnificent in this project. There's actually several big community type platforms on rails, and together I think we're going to solve a lot of these problems together. It's just – I couldn't be happier.

[0:53:18.5] JM: What are the conflicts with rails, or the problems with rails that you'd like to see addressed in the coming years?

[0:53:26.4] BH: Yeah, not a lot of problems necessarily. I think the issues a little bit now is that because node and Go and some other ecosystems have become so popular, that there's just not a lot of – it used to be the case in the Ruby community that everything had a Ruby library wrapper, because Ruby was so popular in web development for so long. Now that it's no longer the case, you just can't bank on that as easily.

Some of the old libraries are getting a little, like some things are just becoming less and less maintained because it's gone from being a super popular, like everybody's starting with Ruby to being like oh yeah, it's got a lot of awesomely maintained projects, but it doesn't have that gigantic community anymore.

Then just running Ruby on your computer, I think is just more annoying than some other things. That's always been the case and it hasn't gotten worse, but some other things have gotten better and that's the difference. I'm personally – I would love to get our project just more tightly configured with some contain – the container ecosystem and stuff, just so we don't necessarily have to think about some of the underlying issues and then everything can run in that consistent environment.

Everything's been fine for our current process, but as we scale, I just really want to make the developer experience much easier, much more simple, much more organized and better than other things that have come before it. I have a pretty good idea of how that's going to happen. I just haven't really had the chance to actually write the code, or really give it all the thought I've

wanted to. Pretty happy with the future. Present has its hang-ups and stuff, but this is how it has always been a very long-term thinking company project, everything. Pretty excited to just keep developing on it.

[0:55:17.5] JM: As we wrap-up, I want to get a sense of what amount of vision you're laying out for the open source community, because I see there's two things you could do. You could just say incremental things to the community and say, “Hey, if you want to get started as a contributor and get involved and contribute code, there's this bug and this little feature and this other little feature, and you can do these things and that'll help you get some traction as a developer.” These new developer type of issues, or you're new to the repo you can get started with these issues.

On the other hand, as a product-centered developer, you want to set a higher level vision, so that people know where you're going and I totally respect the – we've got a lot of different opportunities that we can go in, some of them could make money, some of them might take a while to make money, some of them might be shorter term to getting to money, but some of them might require significant feature development. Now you don't want to get into a situation where you're just developing these big features on your own in the public, in the open-source community.

You want to be able to lay out the vision for the big features that you're developing, so that hopefully the community will help you out in building out those features, or at least they will be aware of those features as you're building them. What's your process for setting out a vision in this open source community, or are you just trying to let people contribute these incremental changes?

[0:56:55.5] BH: Yeah. That whole element is definitely a bit of a work in progress. We have been doing a few things here and there. One pattern that's really been working has been myself, or someone from the core team will build the start of a feature, where it's clear where this is going and then somebody else can run with it. We have a version two of the editor, which just is a bit of an upgrade, make it less buggy, works a little better and is a little bit friendlier.

I hacked that together myself pretty quickly and I got to a point where I was like, “Crap. I'm having a hard time getting this autocomplete to work as it's supposed to. I know these last few things are missing, but I can either keep working on this or I can just let this be up for grabs now.” No longer to the point where somebody has to originate the idea. They get to just finish it based on the obvious way that things have been laid out. That's interesting. It’s like our core team which has a lot more meetings and just knows exactly where they need to be putting their time gets working on certain projects and then depending on the state of things the community can take it from there and then they'll hand it back to us a little bit.

That doesn't really answer the question, I think, but that's where things are now. Luckily, I think we just have a lot of people who are understanding and know that things are a bit of here and there thing. Even with my own team, it's never been my style to lay down a heavy long-term roadmap. It's always been a little bit more incremental, so just we're figuring that out. We're working together. We're all trying to make things work.

One longer-term vision thing that has definitely been – if there's one thing we've laid the vision for long-term, it's the reusability of the platform. That's one thing we really want to get to is that, so people can use this platform for their own purposes. If I want to build a community of musicians, or anything like that, they can build on our platform.

The fundamental difference between us and some of these other build your own social networks is the fact that it's open source and the fact that it's like opinionated and it's not just a general solution. It's got a lot of hardcore thoughts on the way things are done and things like that. We're hoping that that will eventually manifest itself. We're always working towards that in our vision, but not actively maybe on a day-to-day.

We actually do have people. People have come to us already several people, like I want to build this. One person wants to do a book review social network, similar. Some of the other ones out there, which came up and stuff and I have no idea about their business model, or what they want to do with the whole thing, but I'm more than happy to –

[0:59:40.8] JM: Goodreads 2.0, that'll be awesome. I would love Goodreads 2.0.

[0:59:44.6] BH: Yeah. I don't really know where they're going with this, but they want to build it on our platform and they asked about how that goes down. I told them like, “Look, it's going to be a little while before that's going to be a good thing.” The word dev.to appears in the codebase a 160 times. It's not a variable. Those letters are in the codebase a lot, and as well as the about page, it's written into the codebase and everything else.

The fundamental architecture is just a basic application, but it's going to take a little bit of work for people to change the logo, make that a variable. Then even when that all happens, people are going to be, have to be very in tight – in very tight communication with the core team, in terms of upgrades and bug patches and stuff like that. That's a long-term thing, but it's really important for us to keep making progress towards that goal, because this is an open source project.

As much as people have been very excited to contribute forward a lot of different reasons, you don't want to rest on that idea alone. We want to give people a true personal greedy if they care to be so incentive to take part in what we're building. That's just one of these paths. We don't spend all our development time in this, but I am actively laying down the vision in that regard.

[1:01:03.2] JM: Yeah, I like to build your own social network idea, or spin up your own social network idea, like one click and then enter in some config for what your social network is around, and you've got a social network. That would be cool.

[1:01:17.1] BH: Yeah. BuddyPress on WordPress is one version of this and there's a lot of other things, but there's a lot of reasons for building something that's only for a social network, but it's also open source, so that's the difference. Among the open source things, of course, Mastodon I think is an option in some capacity, but it's a lot different from what we're doing. Then a lot of organizations built around this, just our traditional closed source ecosystems, which are not necessarily going to thrive in the same way we think our platform will. It's a little bit of all these factors.

[1:01:48.0] JM: Ben, it’s been really great talking. I'm really happy to see the developments on the DEV Community and happy to see where you've come from three years ago, hanging out and just ideating. You've put those ideas into practice, which I really respect.

[1:02:03.2] BH: Cool. Yeah, thanks so much for having me. It's always a pleasure.

[END OF INTERVIEW]

[1:02:08.8] JM: Today's episode of Software Engineering Daily is sponsored by Datadog. With infrastructure monitoring, distributed tracing and now logging, Datadog provides end-to-end visibility into the health and performance of modern applications. Datadog's distributed tracing and APM generates detailed flame graphs from real requests, enabling you to visualize how requests propagate through your distributed infrastructure.

See which services, or calls are generating errors, or contributing to overall latency, so you can troubleshoot faster, or identify opportunities for performance optimization. Start monitoring your applications with a free trial and Datadog will send you a free t-shirt. Go to softwareengineeringdaily.com/datadog and learn more, as well as get that free t-shirt. That's softwareengineeringdaily.com/datadog.

[END]