Hello, and welcome to Sustain, podcast where we talk about open source sustainability for the long haul. Who are we? Where do we come from? Where are we going? Why did I use a single apostrophe instead of a double apostrophe in my Jason Schema? Oh God. Oh God. Very excited
to talk to you today. I am your host again, Richard Littower, calling from beautiful Montpelier Vermont. And I'm here today with Ben Hutton, calling from across the pond in some UK. Ben, how are you doing today?
I'm all right. Thank you, Richard. Thank you so much for having me today. I'm really excited for coming on. See what discussion goes.
Yeah, this is great. So Ben, you are one of the privileged few. You're one of the people who gets paid to work on open source all the time. You work as postman on the Jason Schema as a specification lead. Can you tell me what that job entails?
Yeah, sure. So Jason Schema is a standard for validating Jason data. Initially, when I started postman, I was kind of crossing off a lot of the stuff that I got left over from working in it. Part-time outside of working hours. At the moment, I'm mostly sort of doing admin work, leading, finding new ways to be forward, trying to gather visions and roadmaps and that kind of thing. So yeah, just trying to move everything forward. So immediate question for a product that is a lot easier to understand the market value for, let's talk about an app or something that has users. It's like, cool, we want to get more users. We want them to click on more ads or buy more subscriptions. Very easy to think about, is this feature part of that or not part of that? For a Schema, Jason JavaScript object notation, believe is that right, is just a way of making documents that's now very popular and everyone uses it. So I don't understand, A, what's to improve? And B, what do you mean by moving forward for the Schema?
Yeah, so Jason Schema has been around a long time. I think the initial one was back in like 2013 or 2014 or something around that time. It was initially a very simple idea of just defining what the properties are like and what sort of types they were. But that's evolved over time to a case of more use cases. And there was a period where it didn't change very much as a specification and loads of tooling built up. And fast forward a number of years and we have this whole ecosystem where Jason Schema is being used way more than validation. It's being used to generate code, it's being used to generate forms, being used to even generate database structures now, which is way more than it was ever intended. It's a real challenge, because in order to do things with something where it wasn't designed, you have to make assumptions. And people make different assumptions and end up with making it not interoperable anymore. That's really the strength of Jason Schema is its interoperability, meaning the same thing across many programming languages, validate something in one programming language, and it validates the same in another. And that's kind of been the biggest value proposition for Jason Schema.
Cool. So that's for Jason Schema. Can you tell me why Postman funds you to work on this?
Yeah. So Jason Schema is used by the open API specification, which is a standard for defining the interface and data structure of APIs. And that's one of the things that Postman uses as part of their products. And Jason Schema is becoming increasingly part of open API and other standards that's underneath it, such as ACE and KPI. Yeah, I think Postman are seeing that Jason Schema needs more work on it. And they're seeing that group that are trying to be forward with that, but really struggling to commit the time and lots of things being stacked up. And it's not being able to be forward at a rate that's really going to see value anytime soon. So while we

will kind of move along very slowly, there's definitely a lot more we could do going full time on that. And not just with the specification itself, but raising the ecosystem and the community up as well.

Talk about community. How many maintainers are working on Jason Schema? How many people in the larger community?

It's really hard to put numbers on that. In terms of the core maintainership, there's only about five or six of us. We're not all full time. So some of us are only there. Some of the times that makes it really challenging. But we're looking at ways we can improve on that number and be forward to have a more sustainable grief of individuals running that team. In terms of the community, across our Slack server, we have five and a half thousand members. Across our whole community, including GitHub and Stack Overflow and Twitter, the estimates come in 15,000 members. It's varying accuracy.

Plus, including every single JavaScript developer ever who will yell at you, things go wrong, right? Yeah.

So very, very large community indeed. It's really cool to see that many people jump in to try to fix the schema. They all know something that I clearly don't, because I'm still a bit confused on why do open standards matter? Like, why is it really important that this is there? Why can't everyone just be like, Yeah, that's it. It exists. We don't need to change it anymore. What are they all trying to get out of it?

You can also tell me this is a dumb question.

No, no, it's a good question. I think there are multiple layers in which I can answer that. Like, there are people that come into Slack server in our community to ask questions. And that's the vast majority of people that have questions. How do I use JSON schema to do this? So how do I use JSON schema to do that? And we have people in our team pretty much in every time zone. So people always get good answers quite quickly. The second sort of layer to that answer is, because we don't hold any implementations of the specification in our organization, and we don't hold any code that people go and run from our specification. There's quite a large ecosystem. So there's at least one implementation in almost all languages. More often than not, there are several and some there are like 10 implementations to choose from. So not only do we have people coming that are using JSON schema just somewhere, but we also have many of the implementers come and ask questions like, Oh, how do I implement this particular change that's come in this last version? Or what considerations do we need to have? Well, how can I improve my library to better support my users? And we get a lot of them as well. We do actually get people coming in who are integrating into larger systems, such as IBM and Oracle, when they're integrating it into their products, they come and ask questions as well. So we kind of support big industry in that sense as well. Big corporations come in and ask questions for us. So one way to look at it would be if I was a company that was doing implementation in Elm or Rust or something, so that I could use JSON coming from a website for my services. And I don't understand how it's exactly specified. I can come and ask you, Hey, how do I implement this in this specification? Is this normal? And if there's something that I need, I can say, Hey, can we have a proposal? Here's an RFP or something to change JSON so that it also works for my use case. Does that sound about right?

Yeah, definitely. I think so. So if we think about it in terms of where companies get value, one of the interesting case studies I did recently was with a company called Six River Systems, which also names Shopify robotics now as they got acquired by Shopify. I think something else went on in terms of them getting sold on somewhere else. We did a case study with them. And one of their biggest challenges was communication between people of different expertise. They had development teams and they had business analysts, which were doing these warehouse analysis and machine learning algorithms to calculate the best way to organize and automate robotics inside fulfillment warehouses. And these teams really struggled to communicate the structure of the data they had and how they were going to work together. And by using JSON schema, they now had a common language to define their data structures. So there wasn't any ambiguity. There's always challenges when you're trying to define data using rows or standard English. If you add contractors from around different worlds, particularly with remote working now, that's increasingly difficult when English isn't the first language. So having a common language to define your data structures, what's allowed and what's not, removing that ambiguity really solves a lot of problems and prevents people putting bugs into production systems. Websites, maybe not a huge deal. Robots, potentially much bigger.

Cool. Thanks for sharing that use case. Makes more sense to me now. That's super awesome. So it's clear to me that you have a framework for working on this and that you have a core team. That's awesome that you have so many members in the community as well who are interested in helping out and assuming not just asking questions, but also diving in and getting their hands dirty with curly brackets and double quotes. This is all part of a larger part of the internet, which is just standards in general, which is often something that we don't think about when we're just using a website. Why would we? It's down the stack. I don't think about my computer running chips every single time I open it. I think, where is my latest Instagram feed?

So can you tell me a bit about why open standards are important?

Yeah, I think that's a really good question. As you say, standards out there and being used, and we often don't see behind the curtain as to how they're created and how they develop over time. Something that's more closer to high end that I use as an analogy for me is JavaScript and programming languages. JavaScript specifically, because I speak behind the curtain a little bit and looked at how that's developed and the processes behind getting something added to that standard and seeing that process. And actually, it's really difficult to make that move forward. And the processes that are involved in these standards matter because it's not just about creating value, but also creating value that people are going to use and be able to use.

It's no good adding something to a standard if a particular company like Microsoft or Google are not going to add it to their browser. And it's taking a lot of effort to get to that point, particularly when we have server-side JavaScript in the form of Node now, and there's a lot of disparity between Node and JavaScript, actually, but the standards that allow you to use something in one place and something else in another and have them work in the same way are increasingly important. And standards are really important for interoperability. I don't know if it's the same as the US, but in the UK, for example, there's standards come in, they recently called Open Banking, and it's enabled intercommunication between banks where people authorize their bank

to share their account information with other banks. So where people would previously have to open multiple banking apps or go through very tricky processes to move money around, there's now a authenticated protocol and standard communication processes to do those

kind of things that would otherwise been really hard. And it's created a wave of new digital first banks. That's really interesting where we see this innovation that's come around and been supported by a standard. So interoperability, making sure that there's less fun in the ecosystem, making sure that people can work together, super, super cool. Part of the process of having a standard in the first place and having an open standard is having a home for it. And there's a lot of standards bodies that exist. There's IEEE, there's ISO, there's Linux Foundation Standards. Where is JSON schema housed and why? Yeah, that's a really good topic of discussion right now in the JSON schema organization. Originally, it was created as a personal draft in the IETF. Personal draft is something that any individual can publish. They can publish revisions as well, but they aim to potentially get it into a working group and become a fully fledged recognized standard. JSON schema continues to be a personal draft for a long time. And the latest version that we're now is still a personal draft. But we haven't expected to move it forward in the IETF process. We had some discussions with various people in IETF and they weren't very receptive. And they even pointed us to our own specification and said, oh, you should join these people over here. They seem to be developing it. So we had some communication issues. And I'm just thinking that what we wanted to do and how we were working didn't really align with how the IETF operates. You know, having said that, we've decided to move away from the IETF process moving forward and publish the standards by ourselves. But we're still taking some of the ethos and ethics from the IETF that they define in their best common practices guide in terms of how we expect to work. We value running code. We value working to consensus. There's definitely value in the ethos and the way that they work, but it doesn't really align with how we want to operate and how we are operating. It's possible for standards to fork. Yeah, we've definitely seen that in the past. We've seen Node.js fork in Node.io and then there was some stuff that happened and
I didn't really keep track of things back then, but that fork doesn't exist anymore, right? And it was kind of blip in history. And those that know the details of that are fresh in their mind. But it's definitely possible for standards to fork. I mean, we can also look at something like Markdown as a standard, which has been forked multiple times. And does it really have a, you know, it's not published by a standards body as far as I'm aware. So there are lots of different flavors, but the general IJ sort of keeps the same. Markdown is a good example because that is used everywhere, even though it doesn't have a standard. Node, I think maybe is a bit different because there's something that you run. You run Node.js and it works and you run IO.js and it worked
till they merged back in together. But when I think of JSON, its ubiquity just seems like it would be very difficult to have another standard of JSON because all of a sudden all the APIs and like every single programmer would have to use shims and make everything work and be very, very hard. What I wonder about that is whether that makes it more difficult to deal with conflict in the community. You just said that you're moving from IETF and you're going to publish your own standard. And I'm curious if that was a really difficult decision and how you manage that as a community lead as someone who's thinking a lot about not just the technical aspect, but the other sides of coding. What is the difference between JSON schema and JSON?
Okay, so JSON is a thing that you make API calls within the data that you receive. That's JSON. So you're familiar with that. You've made API calls. JSON schema is saying you should expect an object and that object will have these properties in and it must have this property

and this property must be a string, for example, like JSON schema is a set of constraints rules that define the structure of the JSON that you're expecting to be somewhere.
What I was worried about was having multiple implementations of JSON schema
such that one person says, no, no, this is valid. The other person says, no, that's not valid,
which I think would cause a lot of difficulties in the ecosystem. And it's one of the reasons
why you can't mess up with standards, right? Yes. Yeah. So does that add pressure? The fact that like, no, it has to work all the time. The schema has to be a single thing. Don't go make another schema where all of a sudden it's okay to have apostrophes that are single instead of double.
How does that work? Yeah. So JSON schema is written in JSON or anything that compiles to
JSON. So people do use YAML, for example, to make JSON schema and their extensions to IDs, then maybe to do that sort of thing. In terms of the JSON schema standards,
occasionally we had people come in and want to radically change things. And they've been too radical at times and we've had to say, sorry, that's not going to work for us. Why do you go and do your own standards? See if you can get any traction from that. And they've done that.
And they've gone through the IDF process and they've got the other end of the standard,
but they don't have many implementations or they don't have a community. And it's kind of a very different position than JSON schema, where other standards are kind of published and done. There's not a continual effort to improve and resign them. You know, JSON schema, we're aware that we've added new things fairly recently. I'd say recently is back in like 2019, 2009,
but still fairly recent for us in quotes. But yeah, it's a huge amount of pressure to get it
right. And one of the big challenges is getting that feedback from the community.
It's a challenge to make sure implementations are consistent, because many implementations are actually not consistent. What we tend to find is they implement the basics fairly consistently, otherwise we would have massive interoperability issues. But the more challenging areas of implementation, such as referencing and understanding how to do relative URI resolution,
it can be quite confusing. Probably the most important thing that JSON schema has to combat that is an official test suite. So we have a test suite which can be run in any language because it's written in JSON. And that allows people to check the compliance of their implementation.
And it's been good and it's been really useful. But the challenge we've had with that again is people take a snapshot of that test suite and never go back and update it or don't check for any refinements. So we have a new tool which has come about fairly recently called Bowtie,
which is a meta tester where people can submit their implementations
and have them run against the test suite that everybody else is running. And you can see the results comparatively across many implementations at once. I like that you mentioned the slow work doing this work. Bowtie sounds great. That's awesome. But you said, hey, our last change was really kind of 2019. That was four years ago, which is a while. And you work on this full time.
I've been working on this full time for the last two years. That's worth noting. So this is still relatively new for me, full time. Well, how do you get funding for this sort of work? Because
it's so low level. Like, where does funding come from in the long term for this stuff? Postman obviously is a very special use case. Postman is, actually, you could probably describe what they do better than I do. But as far as I remember, they help you validate JSON. They help you look at packets coming in. And it's like a developer tool that's very, very useful for analyzing
sorts of things. Yeah, I think Postman started off as just basic API client. And they've really evolved that concept. And to cater for more of the enterprise market, they now provide a full API

platform and they can manage the whole API lifecycle. But it's really only the enterprises that are even thinking about and considering those sorts of challenges where they have APIs in life cycles and multiple instances of those happening. So not only can it build tests and Mark's been a little set allows, probably more importantly, allows other people to come in and test your APIs really, really quickly. So particularly when we saw in a Stripe, for example, develop as a platform, you could go and test that really, really quickly and see it working in Stripe's documentation. And I think a lot of companies saw that. And once it had the same thing, but it's really hard to develop. So one of the offerings placement has is you publish your collections, your sets of APIs to run, and being able to make those first API calls really quickly encourages people to stick around and dig in your platform, or your particular API offering.
So that makes sense for Postman to fund JSON schema. I'm curious, who else is working on it who is in front of by Postman and where are they getting when they come?
There are not that many people working on the guts of JSON schema in terms of the specification. In terms of the tooling, there are companies which are concerned with low code tools such as building forms that use JSON schemers. An easy way to do that. And so one of JSON schemers sponsors
on OpenCollective is a company called Retool. And they use it to generate forms because they can pull the JSON schema from existing API definitions and build forms against them, for example, enables their customers to build forms quicker than they would otherwise.
One of our sponsors is Airbnb, but we haven't had discussions with them. I think one of the challenges is we have a small number of sponsors, but we haven't had the discussions with them in terms of where they're using it and what their value is for them. But Microsoft is another big user and we're slowly having conversations with them. And we have a case study pending, which will probably be out by the time this airs on how GitHub is using JSON schema internally. And it's been there so long that they can't even remember the decision process of adding it in. That's one of the really interesting challenges for JSON schemers. We have people that are looking at it and going, oh, it's an IETF person in draft, which means it's not ready to be used in production right now. So we shouldn't be looking at it. And we have other people that have been using it for so long. They have no idea how they got there and they don't really know how to maintain it or understand it because people have moved on. So we have a really broad spectrum of users, which I think is really interesting for projects or a standard that's continuing to be used and evolve. I like the use of the word evolve. It makes me wonder about what are the stages for standards evolution. So you have the first idea that comes from someone on a Tuesday while doing their dishes who's working in a similar field is like, okay, I can do this. Then they get a lot of people in the room. Sometimes it's a corporate room. Sometimes it's a corporate room with multiple
corporations. Hey, let's bang out of standard. Then eventually you have a group that says maybe a website and you have a standard going, then the community gets involved. And it seems like you end in this situation, which you're kind of in, which is we're working on it. It's slow work and we're funded by people who particularly need this schema, but we also kind of work as consultants. We're kind of here because we're specialized and it helps the company to have the most specialized people working on the product so that they know what's going on. What comes after that?
Are you trying to work yourself out of a job by saying, no, it's good. We did it. Great. Let's done. Now we're not working on other stuff. That's really interesting. Think about it from that

perspective. I think Jason's schema has a lot more to validate. If that's not a little bit too meta. We've added some stuff back in 2019. We're still not really sure how that's going to play out and the value people are going to receive from it. We wanted to communicate how to implement some of those new features a lot clearer. And we've seen uptake by a few companies, mainly to do with extensibility. Back when we weren't working on this full-time and probably even now, we really lacked the time to respond to everybody's requests for adding new features or new keywords to do different things to the standard. So we created this extensibility mechanism that would allow people to add their own functionality but in a standards-compliant way. So implementations could support the same definitions of those bits of logic across multiple languages. And we've seen the open API initiative and the open API standard pick that up and use that.

And we've also seen Oracle attempt to pick that up and use that in their SQL standard product. So while it's still quite young and we're still only really seeing people pick that up and play with it, we want to build better connections with industry and potential sponsors, even the implementation developers and maintainers to sort of draw them in and actually get the direct feedback. With the more complex elements of the standard, we don't get lots of feedback other than from new implementers that are really trying to dig in. What do you think the 10-year view is for JSON schema? Like where will it be in 2033?

Wow, the sort of vision for that far out has been seen in a really small way with one of the case studies we did. So I mentioned earlier how there was a period where the JSON schema ecosystem

really bloomed and took on all these new use cases it wasn't designed for. And my personal vision, I know other people have some slightly different visions for JSON schema and work all that out. My personal vision is I'll be able to see JSON schema use throughout the whole stack of development, right the way from inception to defining your data structures and your models, defining what those different domains that like to defining database structures, to defining forms and the validation used across that whole stack where you have data in any way shape or form and it being the single source of treestrap, all of them. So we did a case study with a company called Open Mesa Data where they actually use JSON schema across their entire stack and the person I interview with you can check out the video recording of the case study previously worked for Uber and built up all of that infrastructure and data processing operations using JSON schema and trying to work it in this way but really struggled. And the ecosystem just wasn't quite mature enough at that time. And as it's evolved and developed, it has reached a point where they can use it across the entire stack. I think they got lucky. I think they were fortunate enough to either find tools that made the assumptions they wanted or to be able to add tools or bits of the stack that didn't exist, had the assumptions they wanted or they could define the assumptions they wanted to happen in place of not having the sufficient language to define what they want. So I think they got really lucky but I would like anybody to be able to do that. In order to do that, we have to define how to remove the ambiguity with code generation or form generation or various UI generation, documentation generation. So there's lots of extensions we need to define in JSON schema to be able to cater for those use cases. And the only way that's really going to work is if we can get buy-in from different industries and different enterprises and companies and organizations that rely on those specific areas of technology. And like I mentioned earlier, the status process isn't necessarily just about

coming to consensus and defining something that people agree on, but it's also about gaining buy-in to implementing and using that standard. Anybody can define a standard but if nobody uses it, it's not really working. It's not achieving what it's set out to define. It's not providing any value in terms of interoperability. So when we talk about extensions, you're talking about developers saying we need this use case. It's going to be really cool. We can work with JSON schema to make that happen. Okay, I'll go talk to you and make this extension valid or workable. And they may be paid for their time, which is great. But if they're paid for their time and then this extension is made, none of that money normally goes back to JSON schema. And I'm curious, what are the financial needs for JSON schema that are outside of people being paid during the normal
jobs for this work? Do you feel that people who make money off of knowing the schema and working together with JSON schema should be donating back in some sort of guild tax or something, or a membership fee for being part of the community in the first place and so utilizing its efforts to do good work? I think where JSON schema is now, we're quite happy in terms of the sponsorship. We've got free placement and being able to have several people working full-time. And I think if placement when it's decided they didn't want or need to support that effort anymore, we would probably be able to find other venues that were looking to support those kinds of activities or would go back to full-time work and do it on the side. But we definitely want to make JSON schema more maintainable. One of the objectives we had from a recent summit, on leadership summit in JSON schema, we ended up drawing a roadmap and we want to assess our financial stability. And what does it really cost to run JSON schema? What are our outgoings? In terms of infrastructure, it's relatively little. We host our site on GitHub and we proxy it through Chadflare, but the real cost of that is practically nothing. Other than people's time, there's not really any cost. I think that's one of the interesting things about being a standard as opposed to an open source project or an open source tool that self-hasts. There are definitely potential revenue streams for commercializing side projects. I know some people in the community are doing that and there's some really interesting things developing. But yeah, I don't really know. I think that's something that I'd be interested in exploring and seeing if people want to contribute. I think there's definitely a sense of JSON schema could be supporting and encouraging development of implementations and helping them become more compliant. If it's the ecosystem grows and develops and stabilizes and increases the level of interlockability it has, the people that use JSON schema and rely on JSON schema are going to have their value increased and they're not
necessarily going to know where that money needs spending or even how to allocate and assess the quality of the implementations. That's one of the big strengths of work we're starting to do over the next year is how to be assess implementation quality, how to make sure they're running the test suite. Then if we have more funds, how can we share those funds out between implementations that are
trying to move towards compliance and how do we support and sustain those implementations? They get to fix bugs or have increased users that have problems or issues that they're reading. So there's definitely a sense of we want to support the ecosystem because the standard strength is the ecosystem. We had two implementations and two languages. JSON schema will be pretty useless. One of the things we often get asked is, why use JSON schema when I can just use this validation tool in a specific language?

And we're like, yeah, that's lovely. Please go and use that. But if you want interlockability, if you want flexibility to define something in a way which is going to be used the same across multiple languages and understood by people with different working backgrounds, then JSON schema really is going to provide you a lot of value. Sounds like you have a really clear understanding for the value of JSON schema, which, understandably, makes sense.
Harari's book, Sapiens, had this funny flip where it says that actually, if you look at the history of humanity, we are just vectors for wheat. Wheat is now everywhere and all we've done is made wheat go everywhere. And if you think about it, all we're doing is helping wheat out, which is really, really cool. And I like thinking about that metaphor when I think about software projects and languages and schemas. So if we think about JSON schema, we're helping it go
forward, we're helping it propagate into the world. What are the major threats to a schema besides forking, besides someone saying it was going to use its other validator? Or is that how schemas die? Is there something else? Like, is there a new JSON schema 2.0 that's coming up that's going to take over JSON schema and make everything, you know, have to be redone? There have definitely been other similar projects that have come and gone. And there are other projects that are trying to do something very similar to JSON schema, but are subtly different or focus on a particular use case or have a particular commercial product attached. JSON schema doesn't have those things. I think the biggest threat would be if something were to replace JSON. But then I think we're going to have a lot bigger concerns than JSON schema, if that's what happens. And we could go back to XML,
I don't know. But I think as a community of developers at large, we've tried to move past that and JSON is the most popular because it's the easiest. And if people complain about JSON in general, having too many great marks or being too verbose, too verbose is an often common challenge or a question like why is it so verbose? Why can't we have comments? Why can't we have this?
And sometimes the simplicity is what makes it so powerful. And a similar with JSON schema, JSON schema, you can just use the simplest parts and use the smallest bit that you need.
And I think that's what makes it powerful is it can be really easy to learn. There are some more advanced things in the data sphere, like RDF or link data. And they're really interesting and they're really powerful for science that need to do more complex things like do ontology comparisons
and do some really clever data analysis of ontological data. But they're really hard to learn. So people really can't get much traction with them. And honestly, people see the value that they might provide, but really struggle to get on board with them. There are similar standards in that space that are trying to use link data. But again, it comes back to JSON.
It's the simplest that we have at the moment. So if we have that with anything simpler, that's not YAML, that is people confused because it's a complicated, then yeah, maybe we'll see something different. But right now, as long as JSON is the top of that, I think JSON schema will continue to be the most used data definition standard for a while.
Bill, a strong urge to YLCSV forever and run out of the room, but I won't.
Thank you so much for sharing your thoughts on this. We are running up on time. So I want to ask, where can people learn more about JSON schema? JSON schema is online at jason-schema.org. If you want to join our Slack server, it's just forward slash slack on the end of that.

That's the best place to learn all about our community and what we're doing
and how I'm moving forwards. We have public roadmap on our GitHub at jason-schema-org
as our GitHub organization. So you can find more about us there as well.
Where can people donate Open Collective? On Open Collective, you can find us at jason-schema.
I see that you're hosted both by OSC as well as OpenJS Foundation. Can you tell me which
one is doing fiscal hosting and how that works? Yeah, certainly. JSON schema started the Open
Collective a while back and we joined the OpenJS Foundation later. The OpenJS Foundation takes
on the JSON schema projects and takes the responsibility for the business elements,
including finances. One of the clauses in joining is that the Open Collective
remains outside of the organization officially, is only for the benefits of the maintainers
directly. And we have a disclaimer at the bottom of our Open Collective.
That makes sense. Where can people learn more about you online?
Twitter at reliquestral. That's probably quite hard to work out the pronunciation of, so
feel free to look in the show notes for that. Also, benbutton.me, which you will find in the
inventory. Feel free to reach out if you have any JSON schema questions.
Sweet. Don't leave yet. This is now the time to show where we have Spotlight,
where we highlight other projects, people or things that have helped us out in our career,
or if you're the host and you've done hundreds of these, just a book that you've been reading
recently, which is what I'm going to do today. So my spotlight today is Sir Gawain and The Green
Night, the Tolkien edition. Tolkien was a really bad poet who really liked iambic pentameter and
assonance. And he's just generally horrible. But I rather like this translation a lot. It's really
fun. So if you want a good read, go read Sir Gawain in The Green Night. And if you want to
understand the story a bit more from a different context, if you're not much of a poetry reader,
the movie The Green Night was exceptional. And I highly suggest checking out a very,
very weird film that I really liked. So those are my spotlights today. Ben, what's your spotlight
today? My spotlight today is a small application called Meeting Bar, which sits in the Mac OS
toolbar.
And it shows that the current items are from your calendar to give you quick access to the
drawing meetings. I found it particularly annoying is after I go and open my calendar, find the
meeting link and click on that to join meetings. I might forget a meeting. So having something
always in my toolbar, which shows me what my next event is, my next meeting is really,
really useful. And particularly for staying focused on the current tasks as well.
Can't tell if that's brilliant or horrible. It's one of the two. Cool. All right. I love that. Ben,
thank you so much for coming on. Listeners, hope you enjoyed this program. If you have,
there's a lot of things you can do. One, you can come join the conversation on
discord.sustainosas.org.
You can join us on social media at Sustain OSS pretty much in most of the social medias.
You can also donate on our open collective at Sustain OSS or go to our website at sustainosas.org
or go to the podcast website, podcast.sustainosas.org to read the show notes. You can also send an
email to us podcast.sustainosas.org. There's kind of a theme with the URL you may have noticed,
where we'll be happy to read any complaints or phrases. Feel free to send them along too
or suggestions for future topics or questions you wish we had asked. Very happy to field those.
Also, like this podcast wherever you download it. And if you're accessing this using a phone or
using the web, wherever you're finding this, know that some work somewhere down the stack has

involved Jason Schema. So go to Jason Schema's open collective and donate to them and be grateful and be awesome. Ben, thank you so much for coming on today. I really appreciated it.
Long time fan of your work. And I hope that you continue to do it possibly. And thanks again.
Thank you so much. And thank you for having me today Richard.