EPISODE 1394

[INTRODUCTION]

[00:00:01] KP: InfluxDB is the open source time series database. It's maintained by InfluxData who offer a suite of products that help organizations gain insights from their time series data. In this episode, I interview Zoe Steinkamp, software engineer and developer advocate at InfluxData. We explore some of the common use cases for time series database such as IoT, and some recent announcements, such as the ability to run Flux queries right inside VS Code.

[INTERVIEW]

[00:00:30] KP: Zoe, welcome to Software Engineering Daily.

[00:00:32] ZS: It's great to be here.

[00:00:34] KP: Well, to kick things off, can you tell me a little bit about how you got your start in software?

[00:00:39] ZS: Yeah. So back when I was around 18, 17, in high school, I had to take a class called business basics. It was not about business outside of learning how to possibly write an email. It was actually a coding class. And I kind of fell in love with the idea of just being able to create something. It was very magical. I would write a piece of JavaScript code and it would do something. I honestly couldn't believe my eyes.

So when I was 19, at the time, the boot camps were just coming out. They were very new. It was very much the wild west of the time. And it was either do that or go to a computer science degree at a college. And at the time, I was like, “I'm not sure that I'm suited for this. I'm not sure this is really what I want to do. So let's just go try the bootcamp and see how that goes.”

I went in for Python, because I thought I would be a data scientist. That's originally what I thought would be fun. In the end, I finished my boot camp and got a job as an AngularJS

developer. And not the Angular that you know and love today. The original AngularJS before it was promptly dropped to later years.

[00:01:45] KP: I recall a little bit, yeah. Well, so the frontend is a definitely interesting place to be. What brought you to explore other areas of the tech stack?

[00:01:55] ZS: Honestly, originally, obviously, I started in the backend. And then I moved into the frontend mainly because I wanted a job for being honest. And I didn't mind it at all. I mean, I started in JavaScript. I thought backend would be just more interesting. And first, I mainly just didn't like CSS. I think that was my biggest gripe with being in the frontend. I like coding in JavaScript. I like doing all that. But I don't know, I just can't seem – Me and CSS just can't seem to get along sometimes.

[00:02:19] KP: Gotcha. And what point in your journey did you start learning about InfluxData?

[00:02:23] ZS: So funny enough, InfluxData more like found me. One of the recruiters reached out to me. And at the time, I didn't actually know what a time series database is. I wasn't really in that field. I wasn't in that space, per se. And so I was like, “You know what? I'll give it a shot.” It was a remote job. The company looked really fun. The product looked really interesting. And like I got the gist of what they were doing and what it was for. And when I actually started the job, I was honestly shocked. I was definitely somewhat in over my head, and that I thought I kind of knew what I was into. But no. It took about probably a year or so to really fully understand every single one – Well, I don't know every single piece. Let's not be ridiculous here. A good chunk of the company in what we do in our platform and our tools. And that's actually how I ended up in a developer advocate role, which is where I am now.

[00:03:11] KP: Well, for any listeners who aren't yet familiar with it, what is in InfluxData? Like what's the product do?

[00:03:16] ZS: So our main product that most people know us by is InfluxDB. And a lot of people call us that, and that's fine. We'll respond. But our entire platform actually includes the InfluxDB, which you can just think of as any other database. You put data in, you get data back out. But we also have a lot of our tooling. So that is our tools like Telegraph, which is an ingest

agent. So you can actually – It makes it easier to bring your data in basically. You don't want to have to fight trying to get it in. And then we also have Capacitor, which does our alerts and notification systems. And that just helps you set up. So you know, you care about what's happening in your time series data. Specifically, you probably care about what's happening at specific times. That's normally why it's so important. And that system definitely helps people set up alerts and notifications for when things are happening at certain thresholds.

[00:04:03] KP: So I've heard InfluxData described as a time series database, and I have a pretty good intuition about what that is and the types of queries I might ask of it and things like that. What does it mean to be a time series platform?

[00:04:17] ZS: So when it comes to time series platform, that means that we handle – Oh, I also forgot to mention this actually in the last question. But we also handle the frontend, the UI. So we actually have a graphing library as well. So we handle everything from ingesting the data, storing the data. Like I said, alerts and notifications, and actually visualizing that data. So we have I think about 10 different graphing types, ranging from heat maps, fan charts, gauges. And recently a newest addition would be our maps, which allows you to – It looks like Google Maps. So you can put actual markers down on it.

[00:04:51] KP: So when a developer gets interested in InfluxData as a solution, is it the sort of primary database for their app that they're going to build something database? Something that's essentially a time series product or solution? Or is it sort of an adjunct to what they're already working on?

[00:05:07] ZS: I would say it's probably more it could be your main database for sure. For example, I have a system at home, a project that I like to call the plant buddy project, but it's just IoT devices hooked up to a plant. For that specific use case, for that specific project. I obviously don't need like a relational database to hold my passwords and usernames. That's where that appropriate use case would be. So for a lot of our customers, they obviously have relational databases to store relational data. And we are used more in conjunction.

[00:05:38] KP: Gotcha. So then what's a typical ingest look like? Do I deploy you into my app and send little events or batches? How do most people ship their data?

[00:05:48] ZS: I think most people ship their data in with the Telegraph ingest. So they basically are just making – Or they do our REST API clients. So they're just making an API request to the database and saying, “Hey, I'd like to write this data in.” Or they're saying, “Hey, I'd like to pull this data back out.” Occasionally, you could also do things like changing it or deleting it. But I would say those are like the two major ways that people get the data in. And with Telegraph, we have over, I think, 200 plugins. So like if you're calling from like a Docker server, we have a plug in for that that you just put on your Docker environment, basically.

[00:06:24] KP: Gotcha, interesting. Well, can you tell us a little bit about what options exist once you've ingested that data? Obviously, you're storing the raw data. But there's services on top of that, from what I understand.

[00:06:35] ZS: Yeah. So we have many services on top of that. I'll start with the first one, which is obviously you could just use this as a pure database. You could just literally store your data and then just pull it right back out. We've definitely got those use cases where people are then just monitoring it on their own. Maybe they're using an outside graphing library. Maybe they're just storing it for future use. But the more common thing would be you put your data in and then you use our Flux language, which is a schema querying language to condense most likely. Because a lot of the times when we get data, we get a lot of things, we get a lot of line items that we just don't mind. We don't care about per se. And so you might want to get rid of those. You might want to clean it up. You might want to change it for specific use cases. Like maybe you really only care about like, “Hey, is the server ever reached this threshold that means it's like about to crash?” But I don't care about the rest of the data outside of, obviously, like the server's ID. Like I need to know where it's happening and when it's happening. But I don't need any of the other data sending me that's extra. So with Flux, you could pare that data down. And also, like I said, you could use our visualization too. So maybe you're really just interested in watching the graphs. See how they change in such. That would be where a visualization library would come in.

[00:07:48] KP: Well, obviously data is data, and any industry that has some time series data could use your solution. But I'm curious if you see any common patterns. Is this a go to solution for metrics? Or IoT? What are some themes?

[00:08:01] ZS: Yeah. So we definitely are a go to solution. I would say our biggest two are server monitoring and IoT devices. IoT devices is probably the more growing one, just because the amount of IoT devices in the world to this point is growing, and will forever be. And I would say – So like one use case I can use with one of our clients is the Rolls Royce power system. It's using InfluxDB to improve their like operational efficiency at their industrial engine manufacturing facility. So they're collecting all the sensor data from the engines of the ships, the trains, the planes, other such industrial equipment, and then they're able to actually monitor the performance in real time and identify like trends. And also, the big thing is predicting when maintenance will be needed, which can obviously reduce the expensive engine breakdowns. And actually, were used a lot for that specific service of monitoring IoT devices for either anomalies, or monitoring them to tell you when it's time to fix them.

[00:09:00] KP: Interesting. Anomalies and maintenance are kind of fascinating to me, because I've looked at some of that data. And any single packet of IoT doesn't usually tell you there's an error. I mean, it could, I guess, if there's some sort of is error true flag. But in general, it's more about looking at some pattern like the number of drop packets in a rolling seven days or something along these lines. How can I get InfluxData to help me get a better view of the full grain data I'm shipping?

[00:09:28] ZS: Yeah. So with that, I'd actually like to use a personal example. So with my plant buddy project, I have like a moisture sensor. It basically can help me know if I'm over-watering the plant. And what I've done is I've set up on Influx a notification that says if the watering threshold for my sensor – It's like 80. If it stays above 80 for more than like five hours, it means the plant is probably been over-watered because it's not soaking up that moisture like a root should. And so I'll actually get a notification from that letting me know that, “Hey, I think the plant might be getting over-watered because it's staying wet for a bit too long.” And that can obviously be at a much bigger scale.

And you're right, it still involves humans making the decisions to be like, “Hey, I would expect at this time, in this temperature, the wind turbine should be outputting this amount. If it's not outputting this amount with these three parameters, I'd like to know. I need to see that pattern, because that means there's something wrong.”

[00:10:25] KP: So you'd mentioned predictive maintenance as one of the interesting use cases for deployment. I got to guess that some machine learning solutions. Can you talk a little bit about how and maybe the roles of who came in and did that? Is it a data scientist figuring out how to implement their features? Or I guess, how does the collaboration take place?

[00:10:43] ZS: Let me think about that for a second. I guess it might be a data scientist. It could also honestly be like a maintenance person, like someone who's actually going to do the maintenance. Somebody who has to make the schedule could also be looking at this data. I mean, I know we like to think that all of this has to be done by like a software engineer or somebody who sits at a computer all day, but I actually wouldn't be shocked at this as being looked at by people who are working on like a factory floor who just can look at something and see a pattern, or just be notified of a pattern. I mean, that's the other thing too, I suppose, is you could set us up to recognize the pattern and then just like set a flag almost to be like, “Hey, this machine was flagged as having this too often. So now we think it should go up on the maintenance roster.” If that makes sense.

[00:11:25] KP: Definitely. Well, I’d love to explore a little bit of the developer experience when adopting InfluxData. Maybe we could start with someone scrappy, similar to yourself building a weekend project like plant buddy. What's the quick start that will get me up and going?

[00:11:40] ZS: Yeah. So with that, personally, I really like our cloud platform. I know we're open source. I know you can go ahead and download it. And maybe this is just me being a very visual frontend developer. I'm very guilty of just liking to work in the web. But I love just getting going on the cloud, because it's so easy. You just get in. We actually have a bunch of templates. So you can just take a Flux query, which we will give you. You don't have to write it yourself. And it basically imports a time series file.

Now if you have your own time series data, you can actually just upload that as a CSV. That's what I like to do a lot for my own personal testing, just out of ease. I mean, I could set up a Telegraph agent. And I have done that before with just my computer. It just kind of monitors my computer. And it just prints beautiful data. It's great for testing on the graphs. But for an actual like user who just wants to become familiar, they could upload their own data, they could

visualize it, they could start to set up some alerts and notifications. And they could just get comfortable on the platform. And then from there, they could obviously expand. They could start to put in a Telegraph plugin. They could start to maybe use that data in a phone app or a website. Maybe they want to display it somewhere else or tell other people information.

[00:12:50] KP: Interesting. I talked to a lot of engineers. I don't as often get to talk to developer advocates. I'd love to hear a little bit about what the role is like and maybe there's ways in which you've seen developer feedback that's been able to influence or drive the product in interesting directions.

[00:13:07] ZS: Yeah. So 100%, as a developer advocate, we are here for the developers. That is our main goal. The biggest thing right now my team is focusing on is not just talking about like the new products, obviously, that we're putting out, which some of them do address previous concerns. Like we have an open source GitHub. So if you want, you can always put tickets in there, if you have issues. They actually are normally addressed. We don't just ignore people and just pretend it doesn't exist. And some of them are things where we've decided, “Oh, hey, I think this would be cool to have in our platform. So we should go ahead and build it.” But that's one of the main focuses.

The other main focus is to see pain points. And sometimes, obviously, with a pain point, we can't always develop a solution quickly. We can sometimes give you a workaround. But we're actually pretty good about making that solution and then letting you know when it's deployed and out in the world. One we just recently did actually was our bucket system. One of our use cases, one of our clients, I think they had I want to say like 1000 buckets, but we didn't have a very good pagination. So they couldn't find some of their buckets, which we gave them a workaround. But just this week, I saw it announced that we finally fix the problem, and they were so happy to see that solution. And it was something that was on the docket to an extent. But it's good to get developer feedback, because we didn't know necessarily that was a pain point until they told us. We might have been slightly aware of it. But it's always hard to gauge if it's actually a pain point or if it's just a known slight issue.

[00:14:39] KP: Mm-hmm. Interesting. Well, I think we've talked a lot about what the product does and touched on some use cases. Could you share a few others? Maybe instances or industries that have adopted and the ways in which they're deploying it?

[00:14:51] ZS: Yeah, so one industry I personally always really liked to talk about is our green industry. So we work with a few different – I like to call them green companies. They work in the green energy sphere, etc. We have SunPower Core. They're using InfluxDB to monitor the energy production and sensor health on solar panels. And obviously, with that data, they can track it. And like I said before, they can find anomalies. They can let consumers know, “Hey, I think your panel might be broken. It's not outputting as much as we would expect.” But it also really helps their company make better decisions, like guide their clients better on, “Hey, you're in this area. You're in the zone. The past year, other solar panels in this relative region have produced this amount. So for your business needs, for your home needs, we're going to suggest this many.” And that data gives him the power to make – To tell those people relatively pretty accurate information, because they have it.

And then the other thing is we're also used to monitor wind farms in the Netherlands, and they have tons of IoT devices. They're not actually using our UI interface just because they have so many. And I think with their system, it's actually set up. I saw at once where they are looking for thresholds from all the different IoT devices. So they're looking for things like, “Hey, we just recently had a weather freeze. The wind turbines are outputting 20% less than it should be. And the wind was violent on this day,” or something like that. And it actually turns the wind turbine to this like orange color, which basically means, “Hey, come give me some attention. I need some maintenance.” And I think that's a really cool use cases on green energy and how we're being used there in the IoT space.

[00:16:28] KP: There's definitely no shortage of database choices a Greenfield project might select Why was Influx the right choice in those cases?

[00:16:36] ZS: I think it's probably because we were easy to scale. And we're great at helping people. Like that's the honest truth, is that we have a saying at our company, it's called Time to Awesome, which is basically the idea that we are trying to make developers lives as easy as possible. Our founder actually built us because he was already facing an issue. Like he was

trying to build his own time series database from scratch. And he was realizing that he was not the only one in this painful position. And that's pretty much where InfluxDB came from. And it's always been the idea that our developers should be able to get us up and running. And they can use our full suite of products. Or I don't think we always call them products. I think we call them tools. They can use our full suite of tools. Or they can always use outside tools and plug them in as they need to. But no matter what, we're trying to make it as painless as possible.

[00:17:25] KP: Well, can we get into a few more of the differentiators? What is attractive for most developers when they first hear about Influx?

[00:17:32] ZS: I think probably the first thing that attracts them to Influx is the fact that we're open source, for sure. I think people like to be able to just start building right away. Have most of the like core solution already there available, and of course free. That's always nice. And then I do think some developers are also attracted to our also cloud one as well, because again, it's free, it's easy to get going. And Telegraph is probably, funny enough, one of our more popular tools. It has so many plugins. We have a lot of developers working on it, obviously from outside the company who monitor and take care of those plugins. And so I think a lot of people come to us actually through Telegraph, because they're like, "Oh, I've been using this to send my time series data somewhere else, but I could use it to send it to you guys."

[00:18:16] KP: Interesting. Could you share a few examples of what some of the resources available there?

[00:18:22] ZS: Yeah. So like I said, with Telegraph, we have over 200 plugins. I'd say most of them are more on the I'm going to call it professional serious side. I'd have to actually look up a full list to see them. But we have like Docker, Apache, pretty much all the big server stuff. We're also on Google Cloud. So you can also like monitor your cloud stuff with us on the – I think it's called GCP platform. And then also funny enough, we actually have Telegraph plugins for not so serious things. Like I think we have Battlefield and we have Minecraft. I have to admit, I'm not a big gamer. I don't know what people are monitoring with this. But clearly, they're really popular.

[00:18:59] KP: Very interesting. So a good use case for maybe someone that has some service they operate for time series. I don't know. Maybe they do some particular flavor of forecasting

no one else does. Could they offer a plugin where someone with InfluxData could try out their routine? Is that kind of options that people can build?

[00:19:19] ZS: Yeah. So we actually have this thing called our community templates as well. So with our community templates, we have templates where you can just kind of plug in like a Docker server where we – I think the big thing is we set it up so that within all the visualization is the main big patterns that you might want to see with your Docker servers are the most important information you're looking for when the data comes in. But you could also do that. Like say you already had a plug-in or say you create a plug-in for Telegraph, you could then create a template as the complementary to that plug-in that people could then use inside of our platform.

[00:19:51] KP: Very neat. Well, I'd love to explore some of the visualization aspects as well. I'm thinking of your REST API that you've mentioned and how I would use that in a lot of applications I've developed where I could just kind of shoot off telemetry messages, fire and forget, here's my status, or my observation, or my sensor reading, or whatever. So I know you've got all my raw data. And I guess it might be interesting to look at that bumpy raw data curve. But typically, I want to look at something more aggregated or different views. What kind of tools do I get out of box?

[00:20:23] ZS: So out of box, in our UI, we have two ways of like building out the graphs, per se. You can do it in like our dropdown checkbox interface where you go, “I want data from this bucket. I want to see these couple of lines. And I want to see them maybe in this time frame.” You can actually get relatively granular data by doing that. You can also write a Flux query, which for some people can be a little bit intimidating, I think at first. But luckily, we have tons of examples online to help people get started. And funny enough, if you actually use the UI, and you do the checkboxes in the dropdown way of getting your data displayed, you can actually see a flux query resulted from that, which I always just love. I just love the fact that we do that.

And so out of the box, you can use those two ways. Flux is obviously the more powerful like way to do it. Like with that, you can do a lot more granularization. You can drop data that you don't want to see. You can get more specific about your timeframes and such. But I find, obviously, at the very, very beginning, most people would start on the first one.

[00:21:26] KP: So Flux is your query language. Could you maybe give an analogy to something? Is it SQL-like? Is it kind of JSON, jQuery-like? What's sort of the flavor of it?

[00:21:35] ZS: I find it to be relatively like SQL. Obviously, it's not the exact same. But that is what it reminds me of. Now that being said, obviously, I'm not like a database engineer. I've only written SQL and I've only written Flux a few times in my life. Obviously, I've probably written Flux more than SQL at this point. But that's what it definitely reminds me of.

[00:21:54] KP: What's the experience for a developer that's familiar and very comfortable in SQL adopting the solution? What's the learning curve for Flux for a person like that?

[00:22:03] ZS: I'd say the learning curve for a person like that is probably a lot easier, because they can normally find either a comment, or a document, or a blog where somebody compares the two. To an extent, like they compare, “Hey, you're looking for this kind of function that would be in SQL. This is how you might do it in a more time series/Flux version.

[00:22:22] KP: And if I'm going to build my application, obviously, I can leverage this tool for storing my data, getting aggregations, and maybe some features if I'm doing machine learning. Do I get any other services I might put my app as well? Like the graph features? Could I display a graph to a user-based on influx data in a convenient way?

[00:22:41] ZS: Yeah, so with that, you have a few options. So you could just pull up InfluxDB on the internet. Or possibly, if you're spinning it on your own machine, you could just see the graph there on like a local host or a server that you're hosting it. We actually also offer our graphing library completely like separate, like it's an open source separate library. So you could just download it and put it on like a website. I've actually done that as like a small project for the docks for that just as an example to be like, “Hey, look, you can actually just put this in like your app. You can put this in your website.” And it's already like pre-setup to handle time series data specifically and graph it. So you don't have to do it all yourself.

[00:23:19] KP: Very neat. And if I was exploring that, it could be open sourced in several languages, or maybe there's some wrapper I need. What's the typical flow given that you have

just about any language a developer could be in they could use InfluxDB? If I want to integrate with some of the tools, what are the common patterns you guys see?

[00:23:38] ZS: Yeah. So with our REST API, specifically, we have I think 12 languages. Most of the big ones, JavaScript, Python, I believe it's C++. I'm sorry, I don't know them all. And with our graphing, we can be used in any language. And we actually have examples and I think a few different versions as well on our graphing libraries GitHub page, just so that way, then it makes it easier for people to get going.

[00:24:02] KP: So we talked a little bit about the maybe weekend warrior developer and getting onboarded. What about a small-medium enterprise or large company? Is there an onboarding story to be considered?

[00:24:14] ZS: I'm afraid I'm not – Obviously, I'm not super involved with the onboarding of those kinds of companies. I do know that they probably go through a slightly different process probably because they come to us more in more, I'm going to say, upfront manner. They're coming in and they're being like, "Hey, we want to use you for X, Y, and Z. And we're going to need this much space and etc." They're going to come in with obviously a longer list of requirements that they'll work with somebody from our customer success team to help them get started.

[00:24:41] KP: Gotcha. Well, buy-in can be a challenge for some developers that they maybe want some 100-year established solution before they adopt to that kind of thing. And a certain concern a lot of people have is, "Well, what am I getting in terms of support? If I do some general Google queries, are there going to be Stack Overflow or responses and Doc's pages and all that kind of stuff?" What's the typical ecosystem like for InfluxData?

[00:25:06] ZS: Yeah, we definitely know – I personally know, especially as a developer, I'm super weary of to an extent what I like to call like small startup-y things on GitHub, even if they're open source, because you never know how well supported they're going to be. You never know what the ability to communicate with people is going to be. But luckily, at influx, we are very good. Not only can you go – And like I said, you can make an issue on GitHub, and we can respond to you. But we're also extremely active on our Slack community. Not only does the

developer advocate team monitor and respond to people on there, but so two people from our Telegraph, from our Flux, from our general – Our general engineers actually go on there and help people. And we also do have a support team just dedicated to helping as well. So even if you are using us as an open source, we are there to help you. We're not going to let you float away on the ocean.

[00:25:56] KP: Very cool. Do you have any tips or tricks for how to manage a community like that? I've seen a lot of Slacks get kind of overwhelming when big group emerges.

[00:26:06] ZS: Yeah, I will say that I think we do a good job mainly because of the fact that we have a lot of our own employees monitoring it. I know, personally, I've been in those communities where it's just other community members helping, and that can be less than ideal, I suppose in a way. Sometimes it can be very helpful. But sometimes those other community members, they both respond once and then never respond again, or they obviously don't have a vested interest, per se, and you successfully using the product. Whereas, obviously, employees are much more vested. And they're also more willing to not only help you, but listen, and possibly, they can also make solutions. That's the big thing. A random community member might be able to help you get over your hurdle. But if you gave them feedback, they're not going to – I mean, what are they going to do with the information? Maybe make a GitHub ticket?

[00:26:52] KP: Possibly, I guess, that'd be it. Maybe?

[00:26:54] ZS: Yeah, but that would be about as far as they can go. Whereas, obviously, somebody from Influx, who's helping you, can do a lot more. Their ability to help you is much more vast.

[00:27:04] KP: Absolutely. Well, you'd mentioned the templates, I believe, is a good place to get started. Could you talk a little bit about any of the starting experiences you've seen as the developer advocate? Do people get up and running in an hour, a day? Or what's the typical experience like?

[00:27:20] ZS: I find that most people can get up and running probably in like 30 minutes, I would say, if they've already got either data in mind, or if they're okay with just using like our

example data. If they obviously have to go find it, or they are creating it, that might take a bit longer. At least with my plant buddy system, it took me about two hours, but it was less Influx and more the fact that the IoT device had to like get hooked up with the board, which then had to get hooked up online. And it was just a whole process, basically. But I would find that most people can get going pretty quick. They'll probably get stuck possibly with either writing Flux queries. Maybe the data they're getting in isn't quite what they want. They want to make some adjustments there. So they might ask some questions in the community about that. Or they'll just get a little confused on the graphing. But for the most part, they'll get going and they'll get things set up. And then they just might get stuck in one of those little things. But again, that's why we have the community, is to help people get unstuck from those.

[00:28:15] KP: Yeah, I've been thinking about taking on a personal project. We're getting more and more of these home Internet IoT devices. And we have some of them automated in certain ways. I was thinking I want to take a shot at trying to get everything on to one platform. Influx database sounds like a great place for me to put that. One of the hesitations I have is that if I keep adding devices, and they're extremely chatty, I might start producing a petabyte of data a month and suddenly get a big bill, not know what to do with it, not be able to query it. How does InfluxData scale up when there's a lot of information and maybe I only care about my daily aggregates and things like that?

[00:28:53] ZS: Yeah. So luckily, we allow you. So, per se, you only wanted to know about the past day or the past two days, or maybe even just the past week, you can drop your old data, or you can aggregate it. So I always like to use the weather as an example. You do not care what the weather was last year on November 9th. But you do care what the average weather was last year in November, or maybe even this week. You don't need to get down to the nitty gritty, but you want a snapshot of what it wants to look like. And that might be the same with like if your IoT device at home is monitoring your water consumption, your energy usage, a light bulb. You don't necessarily need to know the exact little details of when the light bulb was on last month. You just want to know the rough idea that light bulb was on for about four hours. And that's the average that it's been rolling so far for the past year.

So you could see an anomaly if your light bulb is on for 400 hours in a single month. And that's something that we definitely help people with. And especially are actually larger at scale

customers, because they are definitely producing petabytes of data and they do not want to store and pay for all that storage of a lot of it, because they're just monitoring things. They just need the real time data. And then the old data can kind of disappear. Or, recently, we are also rolling out the ability to – I'm going to call it storing it deeper. Like storing it in a way that's more compressed. So you can't get it as easy, but you can get it if you really need it.

[00:30:11] KP: Very interesting. From that, do you see any, I don't know, archival policies kind of arising? Are there best practices or standards people should follow in setting these sorts of parameters?

[00:30:23] ZS: Yeah, definitely, I find on our free tier at least, we give you a 30-day retention policy, which is enough for, I would say, most at-home projects, or small projects. And then with our customers, I know we specifically work with them with our support team to make sure that they are deleting the data they don't want so they don't have to store it. Or like I said, soon, I think we'll be able to start archiving it as well.

[00:30:46] KP: Well, are there any other specifics you really want to tell people who are interested in getting started with InfluxDB?

[00:30:52] ZS: I would say one big specific is definitely look over the blogs, the examples. I can assure you that most likely your use case is not new. Someone's probably done it even if it's a scrappy project to a large company. We probably have an example of somebody there doing something very similar or the exact same thing. And that's not a bad thing either. It's great, because you can either pick up where they have, or you can learn something new, or learn from their mistakes as it comes with blogs and such.

[00:31:23] KP: And we've covered a lot of these. But I'm curious if there any other interesting or quirky use cases that people might not expect. Any templates it'd be interesting to explore there?

[00:31:31] ZS: I guess one slightly quirky use case that I've always liked, or actually I'll have two quirky use cases for this, is one of our clients is using us to monitor plants at scale. I always joke that they were the original plant buddy. And with theirs, they are actually monitoring living

plant walls. So if you've ever been in like a fancy office or fancy building, some of them have those giant walls with plants all over them. And those plants actually require a lot of attention. And they're not so simple to take care of. And they actually probably kind of grouchy. And so they actually have an app where you can kind of monitor the plants health. And all their data is aggregated through Influx to help people monitor. They also mainly work in golf courses. But I'm not as thrilled about a golf course as I am about a living wall. So I always think that one's more interesting. And then another use case as well is we're also being used by a client in New York who's growing plants inside. So I'm sure you've probably seen this before of the indoor gardens, like, obviously gardens at scale, like they're growing hundreds of lettuce heads or whatever, and the purple lighting is above them. Well, again, all of those lettuces need to be monitored. You don't want to waste fertilizers, chemicals, water. So all of that system is also being monitored as well.

[00:32:45] KP: Well, Zoe, can you share anything with the listeners about maybe recent releases or what's on the roadmap?

[00:32:51] ZS: Yeah, so one of the recent tools that we've added, which I am super, super excited about is our new VS Code tool. So you can actually connect InfluxDB. And more importantly, you can write and run your Flux queries inside of VS Code. And as a developer, I like to be able to do everything in my IDE. I don't know. I just like to be in that space. If you tell me that a VS Code tool will help me say five minutes, I probably already have it downloaded, I'm that kind of person. But this tool is so great, because you can actually see – After you run the flex query, you can see the data format. You can see the table with your data that's going to be returned from that query, which makes it so much easier to test. And it makes it so much easier to actually start working with that data. Because sometimes – I'm not going to lie, sometimes it can be kind of frustrating when you're pulling data out and you kind of have to like somewhat sometimes guess what its going to look like or you have to look at the JSON that's coming back to you and you have to look for very specific lines. And with this, it's just so much easier. It's a great solution for devs.

And another thing that we've been also working on recently is making improvements to our notebooks interface, which our notebooks is very much geared towards not the single developer, but a group of people working together, coworkers, friends, a group of people. And

with that, it makes collaborating a lot easier. We've recently been adding annotations. So you can actually like put a little annotate on your graphs and be like, “Hey, this is a note of an issue, or this looks weird. We should look into this later.” Like you can just leave a little note on the graphing on the UI to just – So that your coworkers can see it later.

[00:34:28] KP: Very cool. Well, for a developer who thinks this might be a great fit for a project they're starting up, what's the best place to go online to get started?

[00:34:37] ZS: So yeah, if you go to influxdata.com, that'll probably be the best place to get started. From there, you can find the ability to download our open source. Or you can just get started on our cloud product there as well.

[00:34:51] KP: Well, Zoe, thank you so much for taking the time to come on Software Engineering Daily.

[00:34:55] ZS: Of course, it was great to be here.

[END]