Episode Transcript

Are design systems still for people? A conversation with Elyse Holladay

Chris Strahl [00:00:00]:
Hi, welcome to the Design Systems Podcast. This is the place where we explore where design development and product overlap, hear from experts about their experiences and the lessons they've learned along the way, and get insights into the latest trends impacting digital product development and design systems from the people who pioneered the industry.

As always, this podcast is brought to you by Knapsack. Check us out at Knapsack.Cloud. If you want to get in touch with the show, ask some questions, or tell us what you think, send us a message over on LinkedIn. You can find a direct link to our page in the show Notes. We'd love to hear from you.

Chris Strahl [00:00:26]:
Hey, everyone. Welcome to the Design Systems Podcast. I'm your host, Chris Strahl. Today I'm here with Elyse Holladay. Elyse, welcome to the program. Really excited to have you on.

Elyse Holladay [00:00:33]:
I'm excited, too. It's been a long time coming.

Chris Strahl [00:00:35]:
Yeah, I mean, definitely. I think that we have flirted with the idea of being on each other's podcasts now for like a year, and so I'm glad that we finally got to make it happen. Super excited. We know each other through a variety of community events and just being kind of a part of the design system ecosystem for a while. I'm really excited to talk to you because we have this super interesting kind of hook for today's episode. Are design systems still for people? But before we dive into that, two quick things.

First, if you follow Knapsack and you're interested in coming to one of our Patterns events, they are events for design leaders that want to spend time talking with each other, getting to know each other, networking with the community. And these are things that we hold in major cities all across the country. There's one coming to a place near you if you want to learn more about it. Check out Knapsack.cloud/patterns. You can apply there. Take a look at all of our events, our dates, everything like that.

Chris Strahl [00:01:20]:
Likewise, we're looking for some more suggestions for topics on the podcast. We're actually going to move to a slightly new format here pretty soon. We're going to be pretty thematic about this podcast and go through sort of like seasons and series in a similar way to at least this podcast.

As a part of that, we're looking for people to propose ideas or topics. If you have any and you want to let me know about it, feel free to hit me up in the design System Slack on LinkedIn or just send a message to dsp@knapsack.cloud.

Chris Strahl [00:01:46]:
Without further ado, Elyse, tell us a little bit more about yourself.

Elyse Holladay [00:02:00]:
I have been around the design system space for a really long time. I like to say, since before design systems were design systems. And I know I'm not alone in that. But it's been really interesting to see design systems grow. And so I love to think back and think about that history and the way that it's kind of changed over time. So I started doing CSS architecture back in the day and then kind of like built proto design systems, then kind of moved into what we think of as the current design system era. I was tapped to start the first Design system at Indeed.com, so we were building the system that helped the engineering teams move off of Java to React, which was really exciting. I've held basically every design system role.

Elyse Holladay [00:02:43]:
I've worked on the design side, I've worked as an engineer, I've held the Unicorn design system PM title. I've done training, I've done public speaking. You may have seen me at Clarity or CSS Conf. And in the last couple years I took a break from tech and I left and started my own business. I did a personal style coaching program, so got some entrepreneurship experience and I am back now in design systems at a company called Color Health. I'm our staff design engineer. So I manage our design system as a solo design system team, both on the design and engineering side, supporting both our designers and engineers, and recently started, as Chris mentioned, a design system podcast called On Theme. You can check out the first season.

Elyse Holladay [00:03:25]:
We just wrapped in May. I'll be back with more very soon.

Chris Strahl [00:03:29]:
Awesome. Yeah, and it's definitely worth a listen. I've gotten through about a third of the episodes myself. I love the content. It's great stuff. Thinking about your career arc, you've kind of gone through this in a couple of different iterations, from big to small to entrepreneurship. What's made you kind of like land in the place that you've landed now? I mean, you're obviously like enthusiastic and excited about the work that you're doing, and we're going to talk a lot about that today. But I think that you've kind of followed innovation is maybe a safe way to put it.

Chris Strahl [00:03:57]:
What's got you excited about what you're doing right now and kind of tell me a little bit more about how that career path got shaped.

Elyse Holladay [00:04:03]:
I love the idea of following innovation. I don't know if that's a phrase that I would have said particularly, but I think it kind of is true. I don't really consider myself fully a designer or fully an engineer. I like making things happen. I'm an activator. I like making things go. I like influencing people. So this design system space to me has always been about, how can we do this better? How can we make this thing happen? Like, why can't we have that nice thing? Why can't it be like that? I really got my start in the tech space in general, and my first official real job out of college was at Bazaar Voice, which you may know for popularizing ratings and reviews.

Elyse Holladay [00:04:48]:
So if you are on a site that has customer ratings and reviews, that was Bazaar Voice back in the day. And I was writing css. It was the beginning of CSS architecture. And that really stemmed from, wait, why are we taking all of the CSS that our Java engineers wrote and having to revise all of it before we could even start styling any given implementation for a customer? And that was like the beginning of design systems. Then it turned into how do we make sticker sheets? How do we make components? Now it's how do we make abstractions, how do we make tools? But I've always seen this work as enabling and influencing other people to have better tools and have better processes so that they can make things. And I think that rather than a focus on the design itself or the components itself or the code itself. And I think that's why I love this space. And I think that's why I've followed, as you say, followed innovation or, like, I'm kind of excited about AI, which we're definitely going to talk about, because it's changing the way that we work and changing our behaviors around work.

Elyse Holladay [00:05:50]:
And that's what I think design systems has always been about.

Chris Strahl [00:05:53]:
Yeah. So I love this connection between the idea of how we're going to be leveraging AI and systems in concert with the evolution of systems thinking generally. You know, you talked about CSS and a lot of those, like, early formative things. Right. I mean, a lot of the design systems group in the community and the whole idea of design tokens came out of like a SaaS group in SF. And so all of these kind of like, humble origins of the idea of systems and how we work better are now leading us to this future where we're looking at AI, we're looking at the systems concepts that we've built, and we're saying, how do these things fit together? And I agree with you. It's a lot about reshaping. And so when we think about that, and we think about that concept of reshaping and innovation tying themselves together, the way I've heard you frame this before is about.

Chris Strahl [00:06:39]:
This is changing our priorities. When we talk about those priorities and how that's shifting. Explain to me what you mean by that. Because it's not like this stuff is going away, it's just morphing a little bit.

Elyse Holladay [00:06:51]:
Yeah. Here's what I think is a straightforward example. In the early days of Design Systems, remember 2010-14, we didn't have a way to share component code. And I throw back to this on my podcast, and I've been talking about this a lot because it's so different than how we think about building products now, that if you weren't there and you're not old and you don't remember, then it's hard to kind of wrap your head around that that was the case. And so a lot of the things that influenced design systems today started by how do we solve this problem of literally just sharing a component? And so so much of what we did was make components. Like, first we made tools so that you could even share component code across instances. Then we were like, oh, how do we abstract this? Then there was react. Now we have props.

Elyse Holladay [00:07:46]:
Now we have something like sketch. Now we have Figma. Like, it was all about layering on these abstractions to share a chunk of something.

Chris Strahl [00:07:55]:
Yeah, the collaboration was the problem. Right. It's not like it was about necessarily ever writing a button. That's not necessarily the hard part. It's like getting everybody to use it.

Elyse Holladay [00:08:02]:
No, but we got really focused on writing a button because the original problem was we can't have a single button and actually share it. And so we built all these tools and abstractions so that we could have a single button component and share it. And then I think we became overly focused on that as the deliverable, because it is what we had been trying to deliver for so long. And it was the mechanism that we had to do that collaboration. But I actually see the change that we're going through right now as a full circle moment where we're coming back to some of the original problems that spawned design systems by saying, wait, how do we collaborate? How do we build UIs? And we have lots of tools to do that with now that we didn't have before. But at the end of the day, we're doing the same thing that we were doing in 2008 or 2010, which is building UIs, building web apps, and the tools have just evolved in such a way that some of our initial problems aren't problems anymore. So we're actually having to go Back to some of the really hard stuff, which is people and organizations and behavior change. And it's not the component that's the hard part now.

Chris Strahl [00:09:11]:
Yeah. So when you think about that idea of coming back to those original problems, I think there's also a context thing that I want to frame this in and that the world of AI and the modern way of building products is putting a pretty tremendous amount of pressure on teams. And there is this drumbeat of do more with less. You know, the surface area of our applications is maybe two orders of magnitude larger than it was in 2008, 2010, 2014. And yet teams aren't necessarily bigger. I think companies have a better understanding of how investment works in terms of product. But it's one of those situations where a smaller team has to go faster than ever before. I kind of want to talk about that, especially relative to your context when it comes to this

idea is like, hey, you're a solo person that is managing this for not the world's biggest company, but certainly a sizable organization.

Chris Strahl [00:10:05]:
How do you think about that context and that problem space overlaid with these problems that we have to solve?

Elyse Holladay [00:10:13]:
I love being a solo design system person. I know that it's not for everybody, but as somebody who is cross functional and kind of a generalist, I feel like I have a lot of autonomy. I think small teams, whether it's one or three or whatever, you have some autonomy and some flexibility that a big team just doesn't have. And in some ways I think that actually makes it easier to do more with less. And I feel like big teams, especially big platform or design system teams, you can get really bogged down in team ceremonies, team process, team politics, team governance, decision making by consensus. And some of that stuff is good. I'm not anti process, but I think the bigger a team gets, you spend a lot of time doing your own inter team process. You should be protective of your time.

Elyse Holladay [00:11:08]:
And I'm very protective of my time. But you get very like, oh, we have to have a governance process for anybody who even wants to speak to us to come to a specific office hours, time slot and go through this 85 step checklist. And all of that is done in the service of protecting your time and your roadmap. And I get that and I respect that. But it often, I think, takes away from some of the things that you can just do and the decisions that you can just make. And when you are solo or small, you have to say no a lot. You know, that it can never be done. I think when you automatically assume you can get more resources, you're like, oh, we could just do more if we had more resources.

Elyse Holladay [00:11:47]:
I know that I'm not getting any more resources. So I'm like, all right, how do I own this? How do I find a way to say yes? How do I find a way to make this work? And so I think you have to bring that energy. Our company leadership is not going do more with less. Just use AI. And I know that there are CEOs out there who are doing that. I think we're very much in the novelty stage of like, oh, we can just not have front end engineers and have the AIs do it. And I know that some people are under that pressure, but I don't think about it as trying to do even more. I'm thinking about it as how do I do the thing that is most relevant? Like, what's the next right thing? The most relevant thing, the thing that is actually going to impact my team, my engineers, my designers.

Elyse Holladay [00:12:30]:
I'm never going to get to everything, so how do I do the most valuable thing? And you just have to be really ruthless about making those decisions because you're never going to get to all of it. We had a design system team of, you know, eight or 10 or 20 companies out there have these

big design system teams. All of them are going, we never get to everything. There's always more to do. I think we just have to recognize that that's just going to be the case.

Chris Strahl [00:12:55]:
Yeah. I also think there was this period of time that may be behind us, maybe not where people put a lot of investment into the polish of the system, making it like a portfolio piece almost. Right. And I think that as design systems teams trend towards looking more like yours versus a 30 person organization all working on building out a system, I think that that idea of like, this is a portfolio piece is starting to slide. There's a reason why we built them that way, right? It was this idea that we need to welcome people, we need to draw them in. We need to have all this empathy that we build towards the system. And yes, it's a tool, but it's also this, basically an art piece. And we can all argue about whether that was navel gazing or what.

Chris Strahl [00:13:38]:
Right? But the ultimate thing is a lot of people built that stuff, right? And a lot of big companies invested millions of dollars in what were ostensibly like showpieces where they had like design.myfortune500.com that became this way of drawing people into the organization, as that is Shifting. And we're seeing much more of a trend towards like the functional bits is who we're building these things for. Changing.

Elyse Holladay [00:14:04]:
I think a lot of those published, polished, fancy, shiny doc sites, we're recognizing now that they were marketing places. The whole like zero interest rate period over the last 10 years kind of ending right after Covid was, we had a lot of money. We could spend that money. We were showing off how cool we were and trying to hire people and be out there. And I don't think that's necessarily bad. I think the recognition is that those were marketing artifacts and may or may not have been the most useful to your engineers and designers. And I'm not saying they're not useful, but they may not be the most useful thing. And I love the way that Dan Maul talks about this.

Elyse Holladay [00:14:49]:
He has a saying where he's talking about this, that your documentation should be a journal of how things get used and an ongoing description of the ways that your components and your designs and your design guidelines are actually being in use. Instead of a button is a clickable element that a user can click to do an action and that stuff. I think the other big change is we're recognizing that that's not actually useful. If you're an engineer or a designer on a product team and you don't know what a button is, the documentation is not going to save you. You have a bigger problem. And so now we're recognizing that the kind of documentation we need, the kind of things that we need to provide to our designers and our engineers, is actually different. It's not an essay on, like, what every component is, but it's actually more of guidelines on how things can be used. And it's more about patterns.

Elyse Holladay [00:15:42]:

I've actually had a bunch of requests over the last month or two on, hey, can we get a refresher course on like, how we build layouts with like, flexbox? That's the kind of stuff that my engineers are asking for. And that's like, related to the design system. But you're not going to find that in a specific component documentation.

Chris Strahl [00:16:00]:
It actually harkens back to this implementation of napsack we actually just did, where one of the things that was really interesting about the way this team used Knapsack is they actually had tabs where they would put Jira tickets that were the origination of those components. And then they would put like, lots of code examples and another tab where they could actually have like, implementations and then links to a portion of the live website where you could See that in action and all of that ability to kind of like create a thoughtful understanding of why it was built, the way it was built and how you actually go about implementing. Not just like the idea of like what it is that was super powerful. It's a very pragmatic approach to getting people to understand what it's for beyond just saying like a tile is a surface.

Elyse Holladay [00:16:45]:
Yeah. I think the other thing that's changed is engineering understanding of these abstractions has changed. We touched on this a few minutes ago about, you know, we didn't have components but. But a lot of times teams were building these design systems. They were building components from scratch and they were building documentation from scratch because it was not familiar that you had a card or an accordion or like how do you compose these things? Or how does this work as a pre built thing that I can use. But now we have material ui, we have shadcn, we have chakra, we have ANT design, we have every design system student or engineering student makes a quote unquote component library or design system as like their portfolio project. Like these are common abstractions now in software development and software design. Everybody knows now.

Elyse Holladay [00:17:36]:
So there's a lot of things that we don't necessarily have to say. And I think you're asking like, oh, why do we, oh, we don't need these doc sites anymore. It's like, yeah, because we need to actually document some different stuff. That stuff was useful. We've changed our expectations have changed, our knowledge as an industry has changed and now we need to talk about different things.

Chris Strahl [00:17:55]:
I mean, I think that's a great segue into the idea of like what's on the horizon. Right. I mean now if we're talking about AI and most of our docs are being read by AI, are we actually making design systems for people anymore? Or is this really just about like the context for robots? Talk about a significantly different need. Right. If I don't need to understand the underlying concepts of how my product is built, I just need to understand how that output functions and make sure that that output functions in a way that's expected for a user. Do I even really care about that deep understanding of all the use cases for my tile or my button?

Elyse Holladay [00:18:32]:
I think you're absolutely right. We are going to be using the artifacts of design systems in a much more abstracted way. I don't think that most engineers who use a design system spend a lot of time reading the guts of the component code. Even before AI, you're like, what's there? How do I use it. Maybe you skim the props. You're just sort of being like, does this do what I want it to do? And maybe you read some of the docs. But a lot of times I think what was happening is engineers were looking at the props that were available in the component and just being like, well, I don't see a prop that's named the thing that I want. It doesn't do what I want.

Elyse Holladay [00:19:09]:
Even if it did, if it didn't do it in some kind of clear way. And I think that that is now an accelerated abstraction. But on the positive side, you don't actually have to understand the guts of the component really deeply because you have this LLM augmented way of asking, does it do the thing that I want? Can I put a picture and connect that to the mcp and it can read the guts of the component code and tell you whether or not that's possible? I think, you know, humans are still going to be responsible for the design guidelines, the design thinking, the ux, the output. We've been having a lot of conversations internally about how, as an engineer, even if you use an LLM, you are still personally responsible for the output of that code that you are shipping into the code base. I think it's definitely changing the way that engineers are going to be using these components. And I think, honestly, in kind of a better way, I think we might actually see less of, well, I didn't know it could do that. So I just, like, went off the rails and did my own thing or, like, edited the component or whatever. Because I think you actually have a way to say, hey, can it do this? And have that be interpreted correctly.

Chris Strahl [00:20:24]:
I think it gets back to the idea of, like, what problem are we trying to solve? I don't think design systems have ever necessarily been good at saying, like, these are the boundaries of what you can and can't do. But it's been a good job of being able to say, like, this is how you should think about something or how you should use something. My co founder Evan's favorite example is like, a design system is going to give you a grid and it's going to give you a hero, but it's never going to tell you to not make a grid of heroes. And so the idea of how AI gets used to interpret these models and these ideas being able to be much more focused on the problem you're trying to solve instead of how it's constructed is, I think, really interesting. And it gets to your very first point about layers of abstraction for so long we've been building abstractions on top of code to try to understand, collaborate and convey the meaning of what we're trying to build. We're really empowering a lot of people with AI to say like, okay, I can actually talk about what I want or what my intention is, using more or less plain English, and then have a result that is closer to me and closer ostensibly to the problem than it's ever been before. And so that, I think is my like, provocative statement around other than like, you know, are we making design systems for robots? The real thing that I want to kind of like hone in on is are we shedding abstractions? Because that's kind of what it feels like to me.

Elyse Holladay [00:21:45]:
I see it as another abstraction layer on top. It's the third or fourth layer on the layer cake. I don't think it's getting rid of any abstractions. Maybe that will change over time. But right now I think that design systems are actually really critical for allowing LLMs to do what you just described, which is have somebody, an engineer, a designer, a pm, go in and in their native language without having to learn how to code, say, hey, I want to build something that looks like this and does this and you click on this thing and it expands and whatever, you get much higher quality output when you have components underneath that. There may be a case where at some point we don't even need that and it can just sort of like make it up. But then you have a maintenance problem. I think those abstractions are still all there.

Chris Strahl [00:22:34]:
This gets into the idea of like, how does cursor work versus how does an actual developer using components work? Right. And I'm a big fan of looking at AI in terms of how do we automate or streamline human based workflows that are already prevalent, that are giving us good results, that have good outputs. Right.

Elyse Holladay [00:22:51]:
That's certainly where we are right now. And that may change at some point, but that's where we are right now.

Chris Strahl [00:22:56]:
So if you're having something that is built by basically just like prompting a agentic code gen tool and it's writing REACT for you versus something that is leveraging a design system to say like, okay, what I'm actually passing is I already have the react. I'm just passing props to it or whatever it is, that is the thing that you're doing to actually make it the way you want. The better output, in my opinion, is still not the wholly new generated thing. It's the ability to use the stuff that's been blessed. That's been understood because of like accessibility and performance and all these other concerns that go into like how we make that thing. The problem to me with a lot of like the agentic tools seems that there's no workflow attached to that. There's nothing that says this is how we go about the process of taking this thing that the LLM wrote and actually making it performant and accessible and all the other things that you need to think about when you write a big piece of software.

Elyse Holladay [00:23:47]:
Well, that's the stuff that I think a design system can very impactfully help abstract away and not just in components. One of the things that I've been doing a lot of experimentation with is how do I provide design guidelines into. In our case it's cursor. And when I say design guidelines, I'm talking about stuff like here's how you build when you use flexbox or CSS Grid. Here's things that you should think about around combining components like here, be sure to look for components like this even if the engineer doesn't say anything. And I actually think that there's a lot that we can do to continue to grow that. Right. Talking about accessibility, you're talking about performance.

Elyse Holladay [00:24:28]:
Here's a little spicy opinion for you though. I actually like how much time do we spend trying to get our engineers to be like, don't forget to think about this and don't forget to think about that and don't forget to look at all of the things that are available to you and be sure to use this best practice pattern. I can literally write a doc that tells the LLM that it has to do it and it will pretty much do it. And yes, it's non deterministic. Yes, you don't always get exactly what you want and yes, you have to be good at prompting. But the kind of guardrails that I've been able to put around UI in our system have been super impactful. I can make it so easy for my engineers to not have to think about all of these little annoying arbitrary consistency ways of using SX or using the theme or doing this or doing that. And some of it's linting, but the computer helps.

Elyse Holladay [00:25:16]:
The computer is right there by their side as they go through and do that. And I think we're actually going to see that expand to more to what you were talking about around much larger patterns. I have a special hate for the material UI grid component, which is just like the worst. It's called Grid, it uses flexbox, it's A conceptual grid that doesn't make any sense with how CSS works. It's so bad that they're deprecating it, but things like that. We built these code abstractions around design concepts that actually made writing the code harder. And I think those things actually are going to get kind of ejected out of the system because the LLM can just write CSS grid or it can think about, oh, we have these design patterns around how we show states or how we show alerts and how we make sure that those alerts are accessible. We have a icon actions that appear on hover pattern in part of our product and that's a pattern that could actually be used in lots of different places.

Elyse Holladay [00:26:21]:
It doesn't necessarily have to use the same component. And right now it is the same component. If you think about design guidelines as just like an algorithm, if you can codify what these design decisions are into a document that the machine can comprehend, I do think that there are some abstractions we're going to remove.

Chris Strahl [00:26:41]:
Hey everyone, I'd like to take a quick break and tell you about Knapsack's leadership summits. Every month we host an exclusive in person event in different cities all around the country, bringing together design, engineering and product leaders. These summits are all about sharing our learning with tailored discussions to address the challenges you're facing. The best part, it's totally free. Head over to Knapsack.cloud/events to find an upcoming summit near you and apply to join us. We'd love to see you there. So isn't that then the design system that document you just described, it probably looks a little unlike the design systems that exist today, where there's a bunch of component examples and patterns and stuff like that. But to me that's the specs, that's the props and the slots.

Chris Strahl [00:27:24]:

There's probably some docs that describe how those specs work and enumerate variations and types and stuff like that. But that's the kind of stuff that the machine cares about. And so the way you described it makes it sound like fundamental infrastructure. It's the way these LLMs are going to work. And I'm curious about how you think about human roles here, because I think you and I think about this like very alike in this regard, where the days of this 30 person team that is building this giant product, we used to call design systems a product all the time. I think that there's still a lot of people that think about it that way. When you ultimately look at the end state of that, that looks a lot More like infrastructure in my mind. And in that infrastructure you have a few people that are on a design systems teams that are about maintaining that infrastructure.

Chris Strahl [00:28:08]:
And then you have a whole bunch of other people that value that infrastructure for how they can use it to create. And so the maximal usage of design systems with AI in the future to me is like there's a handful of people that are out there building the context, the control points, the data that is the thing that feeds that machine. And then everybody else becomes a consumer of that through whatever agentic system you're using to make something.

Elyse Holladay [00:28:37]:
Yes. And I think there's some really cool ways that this can enable everybody to. I'm going to use the word. Just imagine me doing air quotes right now. Contribution. I think contribution in the way that we've talked about it for the last 10 years has been dead. I think that it never really quite materialized in the way that we kind of wrote think pieces on.

Chris Strahl [00:29:04]:
Oh yeah, we were all going to have these mini open source communities that existed inside of our organizations, all that stuff.

Elyse Holladay [00:29:10]:
So like the example I was just giving about these hover actions, that's not actually in our design system, that's in a product, but it is a design UX pattern that we have created for this product and can be used in lots of parts of that product. And if you can think about a designer who doesn't have to know how to code to be able to say, hey, this component or this pattern of actions on hover, icons appearing on hover to do actions, I can actually now go in and the component is the prompt of like, this pattern exists and we want to use it in cases like this, but not cases like that. And these are the available icon slots that it can have. If you don't actually have to go in and write the react component, you maybe now have a really different way of contributing those design patterns and design guidelines back to the system. And I think maybe one of the things that you're getting at when you say oh, is that the system, is the system going to be kind of abstracted away is I think that line between design system components and patterns or components inside the product is going to get a lot blurrier because they're all just building on top of each other. And then it's not so much about owning a specific chunk of code as it is owning the definition, the spec of the options that are available when you use it, how you use it, why you might use it, and then relying on that as the interface for how it gets used. But

Obviously that's making a lot of assumptions about everybody on a team is using the LLM tools and right now it's still very fragmented. And I think that's not necessarily the case.

Elyse Holladay [00:30:58]:
I would be sad, I think, to see a world where we have no ability to define or own those code chunks or patterns. Maybe I'll live to eat those words, but I think that not understanding the output of the thing that we're making is a real failure state of using LLMs safely and successfully.

Chris Strahl [00:31:21]:
When you think about the amount of permutations and the amount of creativity that you can get out of a single design system, I tend to think about it like a deck of cards, right? If every card is a component and there are 52 of them, how many five card hands can I make? And the answer is around 2.5 million. That's not even assuming that there's variations, right? And so, you know, if you're in this situation where you have potentially thousands or tens of thousands of variations of things, there maybe is a future where we don't really think about the craft as much. There's a really interesting quote from Scott Belsky, who used to be CPO of Adobe, where he talks about the idea of like indexing on taste, not skill, because the idea is that LLMs are very quickly eroding the value of skill and replacing it with the value and taste. And so what is the best, you know, of those two and a half million different combinations and when do you.

Elyse Holladay [00:32:13]:
Use them and why would you use them and when do you make those trade offs? I think this is a place that we are going to need to, as design system practitioners own and support. I completely agree about taste because so many times, how many times have you heard, well, we can't do that because it's hard in the code, or we can't put it in the system or there's not time. Those are the kinds of problems that are actually becoming easier and easier and easier. The machines are so fast at very well prompted, well defined refactors, even when they are big and complex, that it's a lot easier to say, oh, we could change that pattern everywhere, frankly, we could change that pattern everywhere even if it's not using a component. We are not that far away from being able to say, hey, go find me every place where we have this thing that should be using the component, but it's not and the pattern is kind of wrong and like refactor them. I've already been working on refactors that are kind of approaching that level of.

Chris Strahl [00:33:15]:
Complexity, we are so close to the idea of like, I have a legacy product that I need to update into a like component derived code base and just like doing that with a couple of buttons, clicks.

Elyse Holladay [00:33:26]:
I think we are very close to that. But where the LLM still can't quite do it is in all of the requirements and context that are needed. And I don't mean context in the kind of context you feed to the LLM. I mean, why did we make this business decision? Why is this feature still in there? Is that even a feature that we use? If you are going to feed an LLM a crappy portally built legacy product, it will do a very good job of faithfully recreating all of your stupid ass decisions

and poor architecture. And completely doing a brand new version requires very, very good understanding of the requirements and the needs and how things fit together as well as all of the code implications and the design implications. And so the tools are getting better at that. We're going to see it happen. But I think that taste and understanding the, I'm going to say business logic and business decisions as a very, very big bucket of why might we do something? What matters to us? Where should we focus? What value does it provide to our team or our users or the business or whatever? And being able to guide the LLM to do those things still matters so, so, so much.

Elyse Holladay [00:34:47]:
I think that especially at the exec level, there's a little bit of mag about you just plug it into the LLM and like it'll just spit out like all of this amazing code and like, wow, look at all this output. Output. Great. Yay, rss.

Chris Strahl [00:34:59]:
Hey, I made an iPhone app. I printed out an LLM and a website.

Elyse Holladay [00:35:02]:
Yeah, yeah, exactly. There's a lot of stuff that ICs rightfully are like, excuse me, right?

Chris Strahl [00:35:10]:
There's a lot of cringe.

Elyse Holladay [00:35:12]:
And it's not just because we're attached to craft as an identity, although I think many people are, it's that we recognize that there's lots of complexities that are not being accurately represented in the idea that just making lots of code output is high value.

Chris Strahl [00:35:27]:
It's also interesting, I tend to think about this also in the context of like, is this just another abstraction? Right. So I know I'm contradicting my earlier statement somewhat intentionally and that like, I don't think about my how my operating system works on my computer. I don't think about how the pixels show up on the screen. I just know that like when I pull up a Zoom call, like that app should work in the context of my screen and in the context of my operating system system. And so when I think about that from like a creating product and LLM point of view, one of the things that I tend to think about there is like, is this just another form of an operating system? An operating system is an abstraction from hardware and LLM is an abstraction from a lot of the software decisions that we've made. And we have similar things that we can draw from. Right. I think that few of them function like AI where they have that like non deterministic bent or lean to them.

Chris Strahl [00:36:15]:
They have a much more like creative concept behind them. But I think that what we're talking

about isn't necessarily unprecedented in the way that we think about building. There's very few people, even though I do know one, that think about like how the web should look in 10.

Elyse Holladay [00:36:28]:
Bit color, like are you going in on your Game Boy? Like we're not doing that.

Chris Strahl [00:36:32]:
Yeah, like all the things that people think about when it comes to like how you actually derive the thing that shows up in the glowing rectangle, be that on a cell phone or be that on a desktop or whatever, there's already a lot that comes before that. It's all about the CSS spec, the browser.

Elyse Holladay [00:36:47]:
Like we don't think about that stuff.

Chris Strahl [00:36:48]:
But somebody does and somebody at that lower level is always building that thing. Are we just defining a new lower level?

Elyse Holladay [00:36:55]:
I mean, yeah, I think so. And I think that there's going to be, as with all of those changes, an adjustment in what we think about as, you know, tech savviness or in the value of those roles. I have a lot of thoughts about the valuation or devaluation of particular skills, css, but every time we go through a change like this, we're not programming in assembly anymore. Now we're using the browser, now we're using JavaScript. I love every time I read a headline that's like we really need some COBOL engineers because we have this giant financial or government code base that's all in COBOL and nobody knows how to pronounce do this anymore. Those skills are not going to fully go away and there's still going to be people involved in all of these steps. And I think it's going to be a very long time before we see the fear hype version of this AI future in which humans are, it's like wall E and humans are just completely out of the picture. I think we've got a ways to go before we have to really worry about that.

Chris Strahl [00:38:03]:
I do want my AI to be a much cuter robot than it is right now, that's for sure. I love this idea about, like, hey, look, what we're doing is we're changing the interaction model, we're changing the interface, we're changing the way that we work with the medium that we're destined for, which is ultimately like, for most of us, HTML and CSS in a web browser, or Swift code in an iPhone or an iOS, there's all these different things that represent the tools of the trade. And I think that those tools are morphing. And kind of what I said before, like, maybe we're defining a new baseline about what people have to really be experts in. But you've got this wonderful point of, like, this isn't about discarding what became before. We have this idea of all these things, all these techniques, all these processes, all these assets that we've built up, those are still useful. And especially like the ways of working and the ways that we like, connect as people that I just would like to touch on kind of as a closing thought, right when it comes to working with all these

LLMs. One of the things that we talked about very early on in this episode was the idea that, like, when we started first talking about componentry and design tools and all these things that we were building, it was all about fundamentally the ability to collaborate with other people and to be able to share those concepts and those ideas.

Chris Strahl [00:39:11]:
I think that there is a world where a lot of these LLM and AI tools are enabling that. And maybe the AI is another person sitting around that table. I don't know. But it's one of those situations where I think it is still about how do we think about collaborate.

Elyse Holladay [00:39:26]:
Yeah, I couldn't agree more. I think somewhere between thing that frustrates me, advice I would love to give to design system practitioners right now, is to be thinking a little bit more creatively about how we can actually use these tools. There has been a lot of freaking out that it is going to replace people's jobs.

Chris Strahl [00:39:48]:
And it is. But here's the thing. It doesn't mean that it doesn't create new jobs or new opportunities.

Elyse Holladay [00:39:53]:
But I think a lot of that is also hype and CEOs, no offense, I know you're a CEO, but like CEOs ways of being like, we overhired and we need to just like cut some people and we're going to pretend it's all about AI, even though it's like kind of actually secretly not.

Chris Strahl [00:40:09]:
I need to do some profit taking. So I'm going to blame this new tech on how I can fire people.

Elyse Holladay [00:40:12]:
Exactly, exactly. But I think we have focused very much on, oh my God, it can write a component. Oh my God, it can write docs. Oh, it can generate some code. And you mentioned earlier thinking about craft as identity. If you're thinking about yourself and your only value to your organization. As I sit here and I type semicolons into the glowing screen and I move pixels around and I make an input and a button, I think that is just such a small way of thinking about what you can actually do. I love the content that folks like John Voss put out where they're talking about, you know, how do we actually be more ethical when we're designing? How do we think about the impacts of the things that we're making? Color is a health tech company.

Elyse Holladay [00:41:01]:
We're in the cancer space. We're thinking about like, how do we engage with the healthcare system in the United States. And components feel pretty far away from that sometimes. But we are really, really focused on that as the identity of our work. And Gina has been saying this since the dawn of design systems, but design systems as people. Design systems is behavioral

change. I really think about design systems work as organizational behavior change. And components don't do that.

Elyse Holladay [00:41:33]:
Components have never done that. We were making components so that we could use them in a way that changed the organization and people's behavior and the way that they worked. And so when you're thinking about how you're using AI around your design system and you're going, well, it can just hallucinate design system guidelines, like a human can do that better, or it can just write some component code and it could be wrong. I feel like that's just such a small way of thinking about what it is that you can do with these tools and what it is that you can bring to your company and to your organization and to your team. There are all kinds of other ways that these tools can let roles be in the code and the design in a way that they never have before. Allow contribution, allow questioning, allow understanding, allow knowledge sharing that are not actually damaging to your role. And I think we get really focused on the things that are very scary. And there's a lot more that can be done.

Elyse Holladay [00:42:39]:
I think if we're a little bit creative and obviously it is going to replace some rules, things are going to change. I'm not trying to say that you shouldn't be worried or afraid that there's not things to be freaked out about. But I think that we can maybe be a little more optimistic than I think sometimes. The discourse in design systems currently has been around this stuff.

Chris Strahl [00:42:58]:
The whole idea of does anybody write in jQuery anymore? Yeah, I mean, sure, there's some people that do, but the reality is that job doesn't really exist anymore. Everybody's role undergoes an evolution. This happens to be a big one. But I love your sort of very pragmatic but also very optimistic take on it. I think that's awesome.

Elyse Holladay [00:43:15]:
Thanks. You know, I started this episode saying, like, I've seen all of this change since the beginning of design systems. And one of the things that I've talked about many, many times on the podcast I give a talk at Clarity about this is like our tools change, our needs and our capabilities change and evolve. We built all these tools to solve our own problems and to make new abstractions. And that's where REACT came from. If you have been working in this space since before react, you remember how big of a shift that was. AI maybe is a little bit bigger than react, but we're still seeing the ways that we do the same thing. Building technology, building software, trying to collaborate, trying to share context between design and engineering.

Elyse Holladay [00:43:58]:
We're just doing it in different ways. And I don't know, I don't want to be afraid of that. I don't want to be afraid every day that I go to work and convinced that I'm not valuable, my work is not valuable. And I think that we can always find ways to do good and to be valuable. And I hope that we can.

Chris Strahl [00:44:12]:
Maybe we can all find ways to follow the curve of innovation.

Elyse Holladay [00:44:14]:
Well, just to not be afraid, you know, like not to get like all emotional and serious. But it is hard to be creative when you are afraid and the world is scary and there's a lot to be stressed out about. But when we can bring some optimism to the table, I think it really opens up a lot of different possibilities.

Chris Strahl [00:44:30]:
Awesome. Well, Elyse, this has been so much fun. Thank you for being on, for finally making this.

Elyse Holladay [00:44:36]:
We finally made it work.

Chris Strahl [00:44:38]:
It's been a great conversation and I really appreciate your insights. Thanks so much for taking the time. We'll put in the show notes where folks can find your podcast, but also if you just want to shout it out, love a little cross promo.

Elyse Holladay [00:44:47]:
Yes, totally would love to have you come and listen. The podcast is called On a Theme Design Systems in Depth. You can find it at designsystemsontheme.com or my LinkedIn we're gonna get into season two, hopefully in, I don't know, next couple of months. So if you wanna be a guest, if you have spicy opinions about Design Systems, send me a message and come be on the show. And then, Chris, maybe we'll have you come on the show, too.

Chris Strahl [00:45:10]:
I'd love that. Well, thank you again.

This has been the Design Systems Podcast. I'm your host, Chris Strahl. Have a great day, everyone.

Hey, everyone. Thanks for listening to another episode of the Design Systems Podcast. If you have any questions, topic suggestions, or wanna share feedback, go ahead and reach out to us on LinkedIn. Our profile is linked in the show notes.

As always, the podcast is brought to you by Knapsack. Check us out at Knapsack.cloud. Have a great day, everyone.