http://ctrlclickcast.com/episodes/web-components

CTRL+CLICK CAST #44 Web Components with John Rogerson

[Music]

Lea Alcantara: You are listening to CTRL+CLICK CAST. We inspect the web for you! Today, John Rogerson returns to the show, this time to talk about web components. I'm your host, Lea Alcantara, and I'm joined by my fab co-host:

Emily Lewis: Emily Lewis!

Lea Alcantara: This episode is brought to you by EllisLab. ExpressionEngine is the web professional's content management system of choice. It's great for designers, great for clients, and has an awesome community and ecosystem. Try the core version for free and see how ExpressionEngine can help you be an Internet hero. Visit ellislab.com/expressionengine for all the details.

Our major sponsor, Pixel & Tonic, would also like all plugin developers to know that the Craft 3 Dev Preview is now out. If you want to get a sense of how things work under the hood for Craft 3, then sign up. The real public beta will be out later this year, but that's not all, Pixel & Tonic is also working on a first-party e-commerce plugin. If you want to know when that is in public beta, visit buildwithcraft.com/commerce. All the links and details will be in our show notes.

[Music ends]

Emily Lewis: Today our friend John Rogerson joins us again, but this time we're not talking about ExpressionEngine … we're going to talk front-end: web components. John is a lead design

technologist with General Electric Software. There he works on GE's web app design system. John has also worked for startups and in higher education.

Welcome to the show, John! I'm glad you could join us again!

John Rogerson: Hey, thanks for having me back on! It's great to be back on the show.

Lea Alcantara: So John, can you tell our listeners a bit more about yourself?

John Rogerson: Sure, I live in the Bay Area, in California. I've been out here for almost three years now. When I'm not working, I just like to hang out with my family. I've got two boys. They're just about six and nine, and we do lots of stuff exploring the wonderful Bay Area.

Lea Alcantara: Especially if you guys are outdoorsy types, there is a lot to do there.

John Rogerson: There is, yeah, and the climate is so nice.

Lea Alcantara: A huge change from Scotland, I would think.

John Rogerson: Well, yeah, I would never have thought I would have been living next to the Pacific Ocean when I was growing up in Scotland.

Emily Lewis: [Laughs]

Lea Alcantara: [Agrees]

John Rogerson: So I'm not complaining.

Emily Lewis: When we last talked to you, I think you were still working in a higher education. When did you make the move to GE?

John Rogerson: Well, being here for just over a year. I came out to California to work for another company like a startup, so it was very different obviously from higher education. It's very different to be working in that kind of corporate situation I guess rather than higher Ed.

Emily Lewis: Has your, not job title, but like what you do evolved? You're working on their web app design system, so you're doing a lot of web *app* development?

John Rogerson: Yeah, I mean, it's a lot more focused now what I do really compared to what I was. I'm more of a generalist before, I guess.

Lea Alcantara: [Agrees]

John Rogerson: When I was working in higher education, it's kind of like Chief Cook (and) Bottle Washer.

Emily Lewis: [Laughs]

Lea Alcantara: Yeah. [Laughs]

John Rogerson: I was even called the webmaster sometimes.

Lea Alcantara: Oh my god. [Laughs]

John Rogerson: [Laughs] Yeah, now there's a lot more focus. Well, actually, I work on maintaining and design system, which is quite interesting, and that's where my exposure to web components has come from.

Emily Lewis: Well, that's a perfect segue, so let's start with the basics, like what are web components? Front-end is sort of my favorite area, but I've intentionally not been paying attention to

web components because I can't quite understand exactly what they are and why I would use them. So please … help convince me. [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: Well, I don't fault you for that because it depends on who you're asking, really, what web components are. There are lots of different interpretations of them. It's still very much an evolving spec, and I would say a lot of opinions still are rendered by what web components are, and especially what they will *become* because it's still an evolving thing. But I think the most important thing about web components is to realize that it is an actual spec. It is a W3C spec. It's not some kind of private enterprise-type thing.

Lea Alcantara: [Agrees]

John Rogerson: So because of that, I think web components are definitely here to stay.

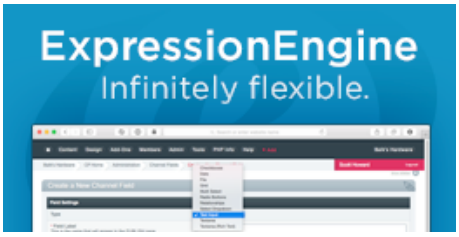
Emily Lewis: [Agrees]

Lea Alcantara: [Agrees]

John Rogerson: What they actually end up becoming when more people are using them and it becomes more of a normal kind of thing in web development is definitely up for grabs and for debate.

Emily Lewis: You shared a good article with me before the show from the, I think, it's Paciello Group, but I'm not sure.

John Rogerson: Yeah, sure.



Emily Lewis: I liked their very high-level definition. It's at the start of this article they wrote about building an accessible disclosure button, and they say that it's a collection of several standards, so these are existing W3C standards:

- Templates
- Shadow DOM
- Custom elements
- Imports

John Rogerson: Yes.

Emily Lewis: And they let us, developers, "create reusable elements and encapsulate the complexity of code and style into components." So can you talk a little bit about this concept of web components as reusable elements, reusable for the whole world? I mean, are they like plugins or add-ons? Is that kind of what it is, like for HTML?

John Rogerson: Yeah, it is kind of. If you're going to make a really crude analogy about web components, you could say they are really iFrames for your own website.

Lea Alcantara: Oh okay.

John Rogerson: So iFrames are usually is to bring in content or something from another website that you can kind of put inside your own. And the problem with iFrames is you have very little control over them, and it's really difficult to kind of, like, penetrate them if you like and do things with them. So you're kind of dependent upon the iFrame being a good actor if you like, yeah?

Emily Lewis: [Agrees]

John Rogerson: And I think another analogy as well for web components that may kind of help people understand them is server-side includes.

Lea Alcantara: [Agrees]

John Rogerson: So they're almost like client-side includes.

Lea Alcantara: Interesting.

John Rogerson: So really, I think as Emily touched on a moment ago, there are four main kinds of parts of web components. And together they form this web component thing. Some of them are really interesting and some of them are kind of like very all over the place, I think.

Lea Alcantara: [Agrees]

John Rogerson: custom elements are probably the easiest kind of thing to understand and put your hat on. So that's just the idea which has been around for a long time, the idea of creating your own HTML elements that a browser can kind of understand and parse and use.

Lea Alcantara: [Agrees]

John Rogerson: I mean, if you think about what the promise of XHTML was, it was that kind of extensibility. And I think because XHTML kind of died a few years ago that web component, well, the custom element part is a kind of like, I guess, an evolution of that part of making HTML and being able to create your own HTML elements.

Lea Alcantara: [Agrees]

Emily Lewis: So let me interrupt you there.

John Rogerson: [Agrees]

Emily Lewis: So I guess I've worked so much in HTML, I know what's available to me and I just make it work. So it doesn't occur to me … what do I need a custom element for?

John Rogerson: Yeah, so really, a good question, and I love HTML. I'm kind of old school. When it comes to HTML, I used to spend a lot of time worrying about "did I use the right element?" and lots of things.

Emily Lewis: [Agrees] [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: And I think, yeah, this is one of the real dangers, if you like, of web components that I think a lot of people interpret web components as kind of almost replacing HTML, "We don't need that old HTML anymore. I can create all these new elements myself."

Lea Alcantara: [Agrees]

John Rogerson: And I think you're seeing a bit of a tension there between people who wants to use web components in terms of custom elements as a way to enhance their HTML, if you like, rather than replace it. And if you think though about the way that we make websites, and especially if you like web apps — that's another show altogether I guess, about the definition of what a web app and what a website is. But when you're making a web app, you can have a lot of what I guess you'd call widgets, which have functionality. And this is a way that you can, for example, like let's just say like an image slider. So using normal HTML, you would create a whole bunch of s and <image> and <figure>s, if you're being really good about things and things like that.

Emily Lewis: Right.

John Rogerson: But with custom elements, you could actually make an element called like, let's say for the sake of argument, <image-slider>.

Lea Alcantara: Right.

John Rogerson: And that could be where you start in terms of making a web component that performs that. I think the really powerful thing about web components is that, first of all, you're making something which can be used kind of almost autonomously, so it could be anywhere on any website that that thing would actually work.

Emily Lewis: If they have all the parts, it would run.

John Rogerson: Yes, if all the parts were fixed together, you could actually use it on any website, and the really powerful thing and really I think the most interesting thing about web components is that kind of idea of if you're producing a kind of web app yourself, to be able to use it within that context and it not kind of have knock-on effects on other parts of the app. But also the other idea is being able to basically use it like a plugin and actually like sell it, I guess, for other people to use. I mean, I don't mean sell it for money, but just being able to make it available for other people, and you're really seeing this in the early adoptions of web components. This is one of the things that, for example, the Polymer Project, they have a lot of like plugins that you can use if you're using Polymer, and we'll probably talk about what Polymer is later, I guess. But yeah, this is another really I think interesting aspect of web components is their independence, but also the way that they can be used anywhere really.

Lea Alcantara: Yeah, like I was looking at the Polymer site, and the first example was a <google-map> element. I guess that's a custom element?

John Rogerson: Yes.

Lea Alcantara: When you use that in that context where it's like, "Okay, a lot of sites use just the custom <embed>, but what if instead of this giant bundle of random code embed, you just have google-map, longitude, latitude and then close tag, and it pulls it?" that seems like really, really clean and easy way to render a map, for example."

John Rogerson: Yeah, I mean, and if you think about what I said earlier about being a real HTML lover, I mean, that's great.

Timestamp: 00:10:08

Lea Alcantara: Yeah, semantics, et cetera.

John Rogerson: Very semantic, yeah, very semantic. And it's that idea of also instead of like relying on someone else's code, that you can create your own HTML. That's a really powerful thing when you think about a good semantic, "I'm going to do a really good job on this and I'm not relying on Google Maps, their embed, and being this a particular way."

Lea Alcantara: Yeah, well, because an embed can be anything. It doesn't have context of exactly what's in there until you start adding all the parameters, et cetera.

John Rogerson: Right.

Emily Lewis: Right, until it's actually rendered later.

Lea Alcantara: Yeah, exactly.

Emily Lewis: And then, this is such a minor thing because validation is not the end all be all. But if ever I have an <embed> from somewhere, I don't validate …

Lea Alcantara: Yeah. [Laughs]

Emily Lewis: Because they include attributes that aren't currently valid. [Laughs]

John Rogerson: Right.

Emily Lewis: It's sort of annoying.

John Rogerson: Right, right.

Emily Lewis: So all right, let's continue talking a little bit about this sort of custom elements. So if you make your own custom element, how are you ensuring that every browser supports it? I mean, we don't even have browsers supporting the specs fully.

John Rogerson: Exactly, yeah. I mean, this is really where things like Polymer and X-Tags come in just now because, yeah, the browser is supporting that. In terms of *native* web components, there isn't really. I think only Chrome, and you could say in Firefox to a different extent are really supporting web components out of the box, if you like. That's really where Polymer comes in.

For example, where it's giving you polyfills, that kind of thing, where it kind of flattens out the browser support. And also what Polymer likes to say is they kind of add a layer of "sugar" on top, and they give you kind of more functionality than the actual native spec does, as well. So I did ask one of the Polymer guys, will Polymer kind of become redundant when web components are fully supported? And his response was, "Well, kind of, but also we'll always going to offer more than what the native spec offers."

Emily Lewis: Right.

John Rogerson: So yeah, definitely, there are a lot of dependencies right now. If you want to use web components, you have to really use Polymer or use X-Tags, which is the Mozilla offering.

Lea Alcantara: Yeah.

John Rogerson: Yeah, it gives you this kind of browser support. Another, you see, because Polymer is a Google thing, there's a lot of momentum behind that. And they are definitely throwing a lot of resources at it, which is interesting because they must be betting that web components is going to be quite a big thing in the future.

Lea Alcantara: [Agrees]

Emily Lewis: So Polymer is a Google effort?

John Rogerson: Yeah, it's an open source project. It's not 1.0 yet, but apparently will be during the summer.

Lea Alcantara: Interesting.

John Rogerson: So it's quite close to being what they would consider something that is production-ready. And the project that I'm working on, we are using Polymer. It's a little bit tricky at times, because the spec and the API from Polymer are changing quite a lot.

Lea Alcantara: Right.

John Rogerson: But I think they are, kind of, really trying to meet this 1.0 target and it should settle down a bit after that. So it can be a little frustrating definitely and confusing because things change, especially around this thing called the Shadow DOM, which is quite hard to get your head around. I guess we could now talk about that.

Emily Lewis: Yeah, that sounds good.

John Rogerson: Yeah, Shadow DOM kind of already exists. It's not a web component only thing. If you think about something like video, the <video> element, which is a really good example. You know you just basically, in HTML5, you put in your <video> element and you pick a source and there might be a couple of other attributes that you can use, but you get an entire video chrome player from that and browsers implement that.

Emily Lewis: [Agrees]

John Rogerson: And actually if you go into your Inspect Element in the browser, you can see a lot of code there that you've not written. That's the kind of Shadow DOM.

Emily Lewis: Wow! That was very helpful. That was actually a really helpful example. Thank you.

Lea Alcantara: Yeah, explanation.

John Rogerson: Right.

Lea Alcantara: Because even like looking at some of these pages and resources that you've given us, the tech-speak, it just sounds hilarious to me, you know?

John Rogerson: [Laughs] Yeah.

Lea Alcantara: It is like, "Web components usher in a new era of web development based on encapsulated and interoperable custom elements that extend HTML itself."

John Rogerson: It's so funny because …

Lea Alcantara: And it's like, "What? Did a robot write this?"

John Rogerson: [Laughs] Yeah, I mean, honestly, I have to ask so many people I work with, what does that word mean?

Emily Lewis: [Laughs]

Lea Alcantara: Yeah.

John Rogerson: Because I'm a very really, really technical person when it comes to the real complex programming side of things. I'm more of your HTML/CSS guy. But the Shadow DOM … I think we've already talked about how web components are sort of autonomous, independent things that you can have many of in one, for the sake of argument, page.

Lea Alcantara: Right.

John Rogerson: And the Shadow DOM is the idea that within those, you actually have completely their own CSS styles. Those styles, because of the enforcement of this thing called the Shadow DOM, mean that those styles don't leak out onto, if you like, parent elements somewhere else on the page and also the other way.

Lea Alcantara: Right.

Emily Lewis: Oh.

John Rogerson: And so you can't actually inherit styles from the rest of the page that will affect what's inside the web component.

Lea Alcantara: Aha!

John Rogerson: Now, there are kind of hacks, if you like … some kind of official, some non-official, to actually kind of circumvent those things. And I do know that Polymer are really having a debate just

now about whether they want to even use the Shadow DOM because of, I think it's really performance-related because browsers are really struggling to use the Shadow DOM, and can they use it in a way that's not crazy to implement. And yeah, really it goes against the grain of how you've been brought up to do web development, if you like. I mean, potentially, imagine, if you have a page like your web app, for example, and you can have 20 or 30 web components easily on there, and each one of those has its own CSS file.

Emily Lewis: So it's just super modular.

John Rogerson: It's super, super modular, yeah.

Emily Lewis: The web is definitely going in that direction on the front-end. I can see this now, yeah.

John Rogerson: Yeah, yeah, and I think also there's another development which is like this HTTP/2 thing. Because one of the things I really care about is performance, right?

Lea Alcantara: Right.

John Rogerson: And one of the things that we've been told many times is one of the biggest performance inhibitors is having multiple HTTP requests. And if you actually think about web components, you're increasing HTTP requests manifold times.

Lea Alcantara: Yeah.

John Rogerson: Because you're having all of these imports. Each web component is importing its own CSS file. And then you're actually having this whole — we haven't touched on the last part of web components which is HTML Imports, where you're actually writing a <link> element like you

would for a style sheet because you're actually using it to import an HTML file, which is a part of your individual web component. You see, things are getting complicated.

Lea Alcantara: Sorry to interrupt you, but that sounds very CMS-like, if you're doing a DRY-type of layout like…

Emily Lewis: Template partials …

Lea Alcantara: Yeah, you've got like your layout template and then you've got your content view template and then you're importing the layout.

John Rogerson: Yeah.

Lea Alcantara: Is that sort of the same concept.

John Rogerson: I think you had a podcast on that already.

Lea Alcantara: Yes, we did with a certain person. [Laughs]

John Rogerson: [Laughs]

Emily Lewis: [Laughs]

John Rogerson: Yes, so I mean, I was thinking about that before I came on the podcast. It's kind of like an extension of that in a weird way, you know?

Lea Alcantara: Yes.

John Rogerson: That concept of actually being very modular, breaking up your logic and the way that you think about how you structure a page and how you put it together is very like the whole Stash approach in ExpressionEngine, for example, which really made that possible in having all your

includes and that type of idea. So yeah, I mean, web components seems to be, as Emily said, that the modular approach to web development is really, I think, something that is the future and it's really here now.

Lea Alcantara: Do you think this is more relevant for the future of web *apps* than it is for, say, just a typical information site or even a new site?

John Rogerson: Sure. Well, you will never want to be prescriptive about this, and yeah, I think definitely that's one of the issues. But there are lots and lots of examples of brochure-type sites, marketing sites where web components wouldn't make any sense. But certainly when you get into whatever a web app is, it would make sense I think definitely to think of a web component as a way to just make your code more manageable. And also, the idea of being able to kind of explore certain widgets. If you think about that in the marketing sense, that's a really quite a powerful marketing thing to be able to like push your web components to other people. And the real thing is, well, there is that built-in idea of web components, it's their extensibility which you can't really do with iFrames.

Emily Lewis: Right.

John Rogerson: So the idea is there that if you wanted to export as a plugin, for example, using that language of web components, someone could actually take it and add enhancements to it and they are not restricted in doing that.

I mean, a really nice example would be like a date-picker where your traditional date-picker is you click in the input field and you get a calendar, et cetera, and you can select. We've all seen really, really poor implementations of that type of interaction, but also really good ones. But you can imagine

the situations where you might want to restrict the date range and all that kind of thing, and being able to allow developers to do is very powerful.

Emily Lewis: Yeah, even I can't say that I know if there's a role for web components for me with a traditional website, But I do know that we already in the process of modularizing what we're building, creating our own basic set of modules that we can pull from for every project moving forward. So if it fits into that, it makes sense I think perhaps for even a traditional marketing type of website.

But one thing that occurs to me, especially as you're talking about, one, you're creating custom elements that meet your individual needs and then, two, you may share that with the community. What if you're not good at what you're doing and you're putting crap out there? [Laughs]

Lea Alcantara: [Laughs]

Timestamp: 00:20:03

John Rogerson: Well, you're gonna have crap web components. [Laughs]

Emily Lewis: Yeah. [Laughs] Is there any … Are there any standards that people are following? Are there any people who are looking for ways to confirm whether something adheres to best practices? Does anyone even know what best practices are?

John Rogerson: I don't.

Emily Lewis: [Laughs]

John Rogerson: With regard to that question, I think the most important thing there is to think about accessibility as a kind of anchor in terms of good practice. That would be my kind of guiding light. How can I make sure that this web component is accessible?

Lea Alcantara: [Agrees]

John Rogerson: Now, obviously, if you're creating custom elements — not even in terms of web components — that is kind of a tricky one for accessibility because how does a screen reader going to interpret, for example, what that custom element is when you just dreamt it up last night?

Now, there are ways around that. You can use the ARIA attributes that are available to kind of create the accessibility enhancements there. The other debate that you're seeing quite a lot is — and I think I already mentioned about — the idea that custom elements are going to replace HTML is a kind of a dangerous one in that respect. That is almost like an anarchy-type, chaotic environment where you're just like, "I have no idea what this guy has done." But actually using web components and custom elements and all of the other niceties as a kind of addition, as an enhancement to existing HTML. And there are ways to do this. I mean, one way to think about this would be using the <button> element, which is kind of an element which isn't really used that well anyway.

Emily Lewis: [Agrees]

John Rogerson: But, if you think about the action that a button can perform, it's kind of like almost infinite, how many things when you press a button can happen, but they're not really part of the actual button — the HTML element — that can really only perform one thing, when you press it and it does something. But that's not kind of an inherent part of the <button> element. But with web components, you can add to the <button> element and make it do certain things that are kind of customized to your particular situation like a dropdown, like some other kind of functionality or interaction that you want to bind to that particular button and you can do that using web components. And I would hope that that would be instead of people writing <my-button>, that they use button as an add on top of it using —

there is a thing called “is” as an actual attribute and that gives you the added customization that you can do on top of just a simple <button> element.

Emily Lewis: I was reading the article you also sent from Jeremy Keith called *Responsible Web Components*, and it’s what you’re talking about. It’s using web components almost as a form of progressive enhancement, using existing HTML that has support by devices, browsers, screen readers and such.

John Rogerson: Yes.

Emily Lewis: And then building on top of it for those systems that can support it, and I agree. To me that makes the most sense than everyone make your own tags. [Laughs]

John Rogerson: Yeah, I mean, progressive enhancement has always been something that I’ve tried to adhere to when I’m making websites. It can be really difficult in the real world if you like, especially when you’re making web apps where almost any web app that I’ve worked on has been very dependent upon JavaScript. And that’s definitely one of the things which web component is true of right now, and it probably will remain so that for the web application to function, it needs JavaScript.

I wouldn’t say that I’m terribly comfortable with that in general. I try to be realistic about it, but yeah, I mean, that’s not just a web component kind of argument. I mean, that’s an overall progressive enhancement argument. But I think, yeah, if you pay attention to people like Jeremy Keith and also the Paciello Group or however you pronounce it, I don’t know … they are really trying to mix or trying to insert the accessibility argument into web components. And I have to say I’ve noticed recently, even in the last few months that accessibility seems to be getting a lot more air-time now, which is actually really, really good because I’m amazed still nowadays how many people just don’t even think

about accessibility as an important part of web development. So it's really heartening to see that it become talked about more. Whether that's part of web components or not is another matter, but yeah, it's really good to see that.

Emily Lewis: Yeah, I mean, I agree. I have noticed it myself, and I'm not even trying to pay attention, but just in terms of what seems to be at the top of people's list where it's not always been at the top. It's always been like the red-headed stepchild you don't pay attention to it until the end, and only if you have the budget or you're in trouble of being sued or something like that.

John Rogerson: Yeah, that's a real shame about accessibility, you know?

Lea Alcantara: [Agrees]

Emily Lewis: Actually, accessibility, maybe because it's got that kind of label, it's really just fundamental coding practices, like good, basic things that you should be following. So let's talk a little bit about what kind of skill set someone needs to have to sort of dive into web components. Do you need to be very comfortable with JavaScript? How much is it writing JavaScript versus writing HTML and sort of I guess building a solution?

John Rogerson: Well, I mean, it depends on whether you're a solo developer I guess or whether you're in a team. I would never say that I was a JavaScript developer. I'd like to be. [Laughs]

Emily Lewis: [Laughs]

John Rogerson: But I think that there's enough work involved to be able to get to a really solid point with web components without having to be necessarily a great JavaScript developer. But the reality is that at some point — especially with ideas like data binding and things like that where you're inserting values into your templates that are dynamically generated through JavaScript, that's not necessarily

an actual web component spec or inherent part of it — but that's the reality of a lot of the web component stuff, especially in the templating side, that you are going to be using JavaScript to kind of provide an API layer as well.

So that's the idea, when I talked about extensibility there, was that you can actually provide using … Polymer gives you a lot of ways to do where you can provide an API layer that you can publish with your web component so you can actually change parameters. Simple parameters, it could be like a visual parameter like how big is the margin or something like this that surrounds that particular pairing element of your component. And you can go all the way to, if you think about something like charting that you can provide parameters as part of an API of an actual web component that would affect the charting capabilities of whatever software you're using to compile the actual charts like within a component, you know?

Emily Lewis: It sounds like it has a lot of potential, and it sounds like if you try to follow best practices that it's probably something you could maybe use today? I mean, it sounds like you're actively using them at your job, but is this something that you think is probably worth just experimenting with at this point or is it ready for like a production-level client work?

John Rogerson: That's the big question.

Emily Lewis: [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: Look, I would say that if your browser support kind of matrix is up-to-date browsers, for example, I would probably say that the last two or three versions of Internet Explorer, Chrome kind of updates itself, Firefox too, that you can really use web components. If you have to support IE9 or

IE8, I would say you probably got more pressing things to kind of worry about than actually worrying about web components because you're dealing there with a lot of the new stuff not being supported. So as far as I know, and this is really interesting, GitHub have made a text editor called Atom, which is a desktop app. And as far as I know that is written entirely in web components using HTML and I think it uses Polymer as well.

Emily Lewis: Oh.

John Rogerson: And that is something which if in terms of as an example of a real world production type thing, I mean, that's really amazing. And I know that some quite prominent companies like, I believe, Salesforce, and they use web components and Polymer to do some of their apps. So I think it really depends on, obviously, your audience.

Just like anything you're doing with web development, it depends on your customers and your users, what they want, what they need and whether web component is going to be a hindrance to that in terms of your development or whether it's an enhancement or not. But I think just from our conversation, you could see the advantages of aspects of web components. Web development, this thing is always kind of evolving and changing, and if you think about the way we make websites now, it's so different to what they were even just five years ago.

The thing that I like about web components more than anything is it is a spec, and I think that's really important. It's not like a JavaScript library, which is kind of just somebody's done that, it's open source, that's great, but it's not really a spec. And I think that's one thing about web components, that if you're going to put your eggs in a basket, you may as well put it in a basket which is sort of approved, and the browser vendors are behind it. I mean, I'm not a Google worshipper or anything like that, but I think just the fact that Google are behind it or putting so much effort into Polymer and

they're very active on the W3C site as well. And this is something which is probably going to be here to stay and will probably influence the sort of future of how we make websites with apps, however you want to think about it.

Lea Alcantara: Well, I have a question, just like a general question, your opinion on this. So when the web started, there is like HTML, CSS and JavaScript and each component there really had like the separation of content, presentation and behavior. And it just seems like, at this point, we're kind of like merging all of them together in terms of that. Do you think that it's rather dangerous or this is just the way the web is moving forward, like making HTML more "programmatical," I guess? I don't know if that's even a real word, but do you know what I mean? Like …

Timestamp: 00:30:16

John Rogerson: I know exactly what you mean, and yeah, I do worry about that definitely. Because when I learned or when I was learning my trade, so to say, I totally dug that whole thing, you know?

Emily Lewis: [Agrees]

John Rogerson: Separation of concerns and it was a very clear delineation, because if you think about, well, it's funny. Actually, if you think about the way HTML started, it had a lot of presentational stuff in it because CSS wasn't really around, so you had to do most of your presentational stuff as attributes in your HTML. We've got rid of that, and then CSS then came, you could do even layout and things like that, and then if you think about the way we use JavaScript, JavaScript was kind of like the last thing that you did.

Emily Lewis: Right.

John Rogerson: So it was kind of like to provide that little bit of like interaction, a little bit of animation and things like that. And now when I come across younger developers, shall we say, most of them are like they don't even think of in terms of HTML, they think in terms of JavaScript.

Emily Lewis: Crazy.

John Rogerson: Everything is JavaScript, and yeah … it is crazy, you know?

Lea Alcantara: [Laughs]

John Rogerson: So that's a really good question, and it's certainly one where I think, yeah, the programmatic part of it is like kind of "Oh dear, please don't take my HTML away."

Emily Lewis: [Laughs]

John Rogerson: But…

Emily Lewis: I think it's one of those things where it's worth maybe for me to experiment a bit, but I think what I'll be really spending my time in is staying up-to-date on what some of the people in our field who I think are good advocates for best practices and standards like Jeremy Keith, for example. His two articles are reasoned, they're balanced, the pros, the cons, the recognition that we don't really know where this is going, so we should be talking about these fundamental issues now to help inform how web components evolve.

John Rogerson: Yeah, Jeremy Keith is obviously a really influential person, and I really like the articles that he writes. And because of what you just said, he's very reasoned and isn't just chasing the latest and greatest, which is definitely a danger with any part of web development, that we just jump on the latest bandwagon and everything, this is going to solve everything. And usually with

things like that, burn and fade away. So yeah, the idea that Jeremy says about the web being a continuum I think is really important in terms of how you think about anything.

Emily Lewis: [Agrees]

John Rogerson: Especially with things like web components, they are kind of like on the cutting edge, if you like. And you know what, I'm not a cutting edge person, but it's kind of cool to be sort of like in the trenches of web components right now because you can see, as you've said yourselves, there are so much potential there. And whether that potential is good or bad, we have to see, but that is up to us in a way, you know?

Emily Lewis: Yeah. I mean…

John Rogerson: As web developers to influence that.

Emily Lewis: Absolutely, and really the only way you can influence it is to put your hands on it, work with it, see how it is, and see what other people are doing with it.

John Rogerson: Yeah.

Emily Lewis: Yeah, in a way it's kind of exciting, if this could be something that could — I don't know if it's possible to reignite my passion for HTML, but just reigniting my passion for thinking about things with that sort of modular approach and that extensible approach.

John Rogerson: Yeah, and I was thinking about the microformats stuff that you are obviously well versed in it, Emily, and just how that was I remember when that first started and came on the scene, that was almost an extensibility of HTML in a way.

Emily Lewis: Yeah.

John Rogerson: That you could use these classes that had actual function rather than just hooks.

Emily Lewis: Yeah.

John Rogerson: And that's kind of where I think web components, I would hope that the spirit of that is part of web components, you know?

Emily Lewis: I agree. I definitely felt some of that. I drew that parallel with microformats and structured data efforts, and sort of making your HTML mean more and do more for you without really fundamentally affecting the HTML itself.

John Rogerson: Yeah, if there's developers out there who's like, "Oh, this web component thing in a sense is way too complicated, and I don't think I could ever find a reason to use it." Well, the Google Maps example is a perfect kind of illustration of how anyone could really use this or see a benefit from it.

Lea Alcantara: Yeah.

John Rogerson: The idea that you could have your own Google Map element is really cool. Because even if you're making a very low-key site for a small client, oftentimes they'll want a map and sort of occupation and this would a way that you could actually enhance that, do your own thing with rather than just plot in the embed and hoping for the best.

Emily Lewis: Right, and hoping that it works and looks good. [Laughs]

John Rogerson: Exactly, yeah.

Lea Alcantara: So you kind of pointed this out a little bit earlier on the performance aspect, because it seems it's like really exciting and you kind of mentioned that a page could potentially even have 30

web components and you're importing everyone's CSS files, and et cetera and so forth. So is there any way to like mitigate that, or like if you need to have these components anyway and you did it the "traditional" way using embeds or third-party services or HTTP whatever … is one faster than the other? Or does it all cancel out?

John Rogerson: Yeah, that's a really good question, and it's one that all kind of people are maybe thinking will be fixed. [Laughs]

Lea Alcantara: Oh okay. [Laughs]

John Rogerson: But you know what, there are two things really to talk about there. The first is there is a new version of HTTP, believe it or not, which is HTTP/2, and basically that means you can have concurrent requests. So it kind of removes the entire problem of having too many requests because it's no longer an issue for the browser, so that's really cool.

But I don't know of anything that uses that yet, and I'm not really sure how you can use it yet. If you're using Polymer, they do provide kind of task called Vulcanize, which I know one of your questions is *Star Trek* or *Star Wars* or something … That is a *Star Trek* reference. You know I have no idea, but that actually kind of munges all of your requests into one.

Lea Alcantara: [Agrees]

John Rogerson: That's what they claim anyway, so you kind of remove that definite problem. Yeah, it's just not feasible to have a 150 CSS file requests on your pages right now.

Lea Alcantara: Right.

John Rogerson: Now, I don't know if HTTP/2 completely removes that as an issue. I think it does? But, yeah, the Polymer project do promise that Vulcanize kind of like munges all of your JavaScript, concatenates everything, minimizes all your CSS so you don't have these multiple — hundreds, potentially — requests for these individual files on your page. So the project that I'm working on, we are going to definitely obviously use Vulcanize.

Lea Alcantara: Right.

John Rogerson: We haven't actually used it yet because depending on what browser you use, there are definitely performance issues.

Emily Lewis: [Agrees]

Lea Alcantara: [Agrees]

John Rogerson: And I think that isn't like something that Polymer is avoiding. They're really working hard on sort of making that better, and I know that they have made a lot of improvements on their kind of most recent versions of Polymer in terms of performance because that was definitely a concern for them … it is for us, there are potential performance issues.

Emily Lewis: Wow! This is really interesting. I'm glad we have this conversation.

Lea Alcantara: [Agrees]

Emily Lewis: I feel like it's the beginning of something that we will definitely be talking more about in the future.

Lea Alcantara: For sure. But before we finish up, we've got our Rapidfire 10 Questions, so our listeners can get to know you a bit better. Are you ready, John?

John Rogerson: I'm ready!

Lea Alcantara: All right, first question, Android or iOS?

John Rogerson: iOS.

Emily Lewis: If you are stranded on a desert island and can only bring three things, what would you bring?

John Rogerson: Can I bring some devices?

Emily Lewis: [Laughs]

Lea Alcantara: [Laughs]

Emily Lewis: You can bring whatever you want, it's your … stranded.

John Rogerson: Okay.

Emily Lewis: [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: I would like to bring something that plays music.

Lea Alcantara: [Agrees]

John Rogerson: I think I would like to bring – could I have my family with me?

Emily Lewis: [Laughs] Sure.

John Rogerson: That would be nice.

Emily Lewis: Family could be one thing. [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: One thing, okay. And I guess some maybe distress flare or something to get rescued. Don't think I'd like to be stuck on a desert island.

Lea Alcantara: Hey, that was kind of like a good mix of practicality and comfort.

John Rogerson: Right, except there's no food around or anything in there.

Emily Lewis: Yeah. [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: Food and water probably are important.

Emily Lewis: I was going to say something totally inappropriate about like cannibalism and that's what your family is for. [Laughs]

Lea Alcantara: [Laughs]

Emily Lewis: It was really wrong. Let's just continue with the next question. [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: [Laughs]

Lea Alcantara: What's your favorite TV show?

John Rogerson: I don't really watch TV so I can't really answer that with any authority. I don't really know what's on TV these days.

Emily Lewis: What about your favorite dessert?

John Rogerson: Oh, now you're asking.

Lea Alcantara: [Laughs]

John Rogerson: I'm going to sound all sophisticated and say crème brulée.

Lea Alcantara: Yummy.

Emily Lewis: Aha! You are our second crème brulée answer. [Laughs]

John Rogerson: Oh, excellent.

Lea Alcantara: What profession other than your own would you like to attempt?

John Rogerson: [Laughs] I'd love to be a train driver.

Lea Alcantara: Train driver?

John Rogerson: Oh yeah.

Lea Alcantara: Like an old choo-choo train, like an old one?

John Rogerson: Oh, I'd love to just be an engineer on freight trains going across America.

Lea Alcantara: Oh, interesting.

Emily Lewis: What about a profession you would not like to try?

John Rogerson: The guy that picks up the roadkill.

Emily Lewis: [Laughs]

Lea Alcantara: What's the latest article or blog post you've read?

John Rogerson: [Laughs] Probably something for the podcast.

Emily Lewis: [Laughs]

Lea Alcantara: [Laughs]

Emily Lewis: If you could have a super power, what would it be?

John Rogerson: Oh, time travel.

Lea Alcantara: What music do you like to work to?

John Rogerson: Anything, I've got pretty eclectic taste in music.

Emily Lewis: Right. Last question, cats or dogs?

John Rogerson: Oh man, that's so tough. I have a cat that behaves like a dog. Is that a good answer?

Emily Lewis: [Laughs]

Lea Alcantara: Yeah.

Emily Lewis: Cat-dog.

Lea Alcantara: Cat-dog.

John Rogerson: It has to be both. Can imagine one over the other.

Lea Alcantara: Fair enough. That's all the time we have for today. Thanks for joining us, John!

John Rogerson: Thanks. It's been great!

Emily Lewis: In case our listeners want to follow up with you, where can they find you online?

John Rogerson: They can find me on Twitter as @iboxifoo. I guess they can go to your website and see how it's spelled.

Emily Lewis: [Laughs]

Lea Alcantara: [Laughs]

John Rogerson: Also, iboxifoo.com is my website.

Emily Lewis: All right, thanks again, John! This was another great conversation!

John Rogerson: Oh, thank you so much! It's been great to be on! I hope people learned something about web components, and if their interest is piqued, they might look into it more.

[Music starts]

Lea Alcantara: Awesome. We'd now like to thank our sponsors for this podcast: EllisLab and Pixel & Tonic.

Emily Lewis: And thanks to our partners: Arcustech, Devot:ee and EE Insider.

Lea Alcantara: We also want to thank our listeners for tuning in! If you want to know more about CTRL+CLICK, make sure you follow us on Twitter @ctrlclickcast or visit our website, ctrlclickcast.com. And if you liked this episode, please give us a review on iTunes, Stitcher or both!

Emily Lewis: Don't forget to tune in to our next episode when we are going to talk about teaching and get some perspectives on teaching UX design with Leslie Jensen-Inman. Be sure to check out our schedule on our site, ctrlclickcast.com/schedule for more upcoming topics.

Lea Alcantara: This is Lea Alcantara …

Emily Lewis: And Emily Lewis …

Lea Alcantara: Signing off for CTRL+CLICK CAST. See you next time!

Emily Lewis: Cheers!

[Music stops]

Timestamp: 00:40:19