ERIC EGGERT: Hello and welcome to my talk, Respecting User Preferences On The Web, at the Future Date conference recorded in April 2020. My name is Eric Eggert, and welcome to my talk, I guess. So I should be in the chat answering questions right now. So if there's anything unclear that I'm saying, just shout out and I will try to answer as soon and as quickly as possible. Let's go in it.

So my name is Eric Eggert, and I work with Knowbility as a web developer and teacher, and I'm also leading the tech team there. And at Knowbility, we want to improve the web for people with disabilities and everyone else. I have been working for W3C and the W3C Web Accessibility Initiative over the last seven years, first in new projects and then on behalf of Knowbility as a Knowbility fellow. And during that time and always, I'm thinking about what can we do for users because users are important.

And I want to start out with some things that I just claim to be right or that I just think about and that we want to take into this talk. And the first thing is that no user is the same. Users differ. They have different needs. They want different things. And we know that especially if we're in the accessibility community, there are a lot of people with a lot of different user needs.

And this is a quote from the Microsoft app developer guidelines, "Designing inclusive software results in improved usability and customer satisfaction for everyone." So you don't need to take my word from it. There's a huge-- there are different needs for different users. And if you make sure that you catch them all, it's even better for you.

But that means that accessibility must be a foundational value for your company and your processes in your everyday life because otherwise it just doesn't work out. Because everything has to fit together to make sure that accessibility is not only something that you want to do at the outset but is something you carry through to the end.

And this is a GIF of Neil Patrick Harris with a stack of papers that he arranges neatly. It says, "Hopes and Dreams" on the cover, and then he's throwing it away behind himself, showing that the hopes and dreams, they are no more. Yeah, integrating accessibility to such a degree-- that is probably a hope, maybe a dream.

And one of the reasons for that is that accessible web development is complicated. And whenever I say that, people are getting defensive and saying, no, it's actually really easy. You can do the simple things, and that helps a lot of people. But to do it really correctly, it takes a different kind of level of knowledge to really do it.

Well, this is the WebAIM survey, and this is not the current number. It's the number from before, which was a little bit more optimistic, that says around 2% of the top 1 million websites have no accessibility issue at all. That's very bad.

I mean, yeah, you can argue if there's one alternative text missing if that's really that bad, but it says a lot that the amount of inaccessibility is just so high. And that's just testing using automated testing, not using expert testers, which will find usually more issues as well.

And there are a couple of key observations in there, like 60% of that 1 million home pages had ARIA present. So although they weren't accessible, someone thought, oh, we should do something for accessibility, and we put ARIA in there. But on the other hand, home pages with ARIA present averaged to 11.2 more detectable errors than pages without ARIA. So was this really an improvement? I don't know.

Pages with valid HTML5 doctype had significantly more errors. That should be more errors in the last sentence. And that makes sense because they probably are more modern websites with more complex interactions. And that creates more accessibility problems, like it has more space for accessibility problems.

So what we can surmise is the more complex a website is, the less accessible it is. And some people try to tame that complexity with even more complexity. So for example, you have something, and then you want to-- and this is a complex widget, so you want to add more stuff to it. And then you add ARIA on top of it, but you don't really understand how ARIA is supposed to work. It makes it even more complex, and it fails in more certain circumstances.

So you're putting on layers upon layers upon layers, but you're improving for some people, but for others, it gets even harder. And that, for some people, they get the impression that they care, and they want to do something, but it just doesn't work out. And that's disengaging for them.

This is the Thirteenth Doctor that says, "I'm too nice. That's what happens when you try to be nice." Yeah, and sometimes that happens. You try to be nice and try to be forthcoming, and you add accessibility features that you think will improve, and then it just doesn't work out in ARIA.

And then there are screen reader differences. And that's why I'm not a big fan of testing with screen readers because different screen readers just have different design principles. And that blew up a couple of years now ago with Scott O'Hara, who found that VoiceOver and Safari remove list element semantics when list-style: none is used. So if a list doesn't look like a list, it's not announced as a list in VoiceOver and Safari. And the whole internet was like, what?

Four GIFs of Doctors that say, "What? What? I'm sorry. What? And what? Hang on a sec." And that's Eccleston, Tennant, and two times Jodie Whittaker. And then we got into discussions on does that make sense or what? And James Craig from Apple answered the bug report and said it was a purposeful change due to rampant "list-itis" by web developers.

Well, that means web developers have used lists to structure content, which usually is a good thing, in a way that overwhelmed users using assistive technology because everything was a list all of a sudden, which doesn't make a lot of sense, right? But there we are. So Apple has this heuristic looking into it and deciding if this list is actually a list.

But what does that mean for me as a web developer implementing lists and Safari not reading it? Well, it doesn't look like a list. It means basically nothing. And to show why it means basically nothing, I have a timeline of events here that I want to show.

This is-- in 2014, Apple did the first commit that removed some list semantics. In 2015, additional code was added just to make sure that they didn't have false positives or things did not get announced as lists that shouldn't be announced as lists.

And then nothing changed for four years. And then in 2019, the very beginning of 2019, the web notices and says, what is going on? And James Craig, again, in the same bug thread, said, "Customers seem much happier in the three years since this change went in." And again, we didn't notice even, and users seemed to be happy, so seems like a win to me.

This comes back to the old question, do all websites need to look the same in every browser? And there is a website about that, and that just displays a "No." And the same is true for screen readers. They don't need to look-- they don't need to produce the same output in every circumstance. I mean, some predictability is nice, but in the end, it's all about the user and their assistive technology. That's the core. They are in the middle of this. It's not us telling the screen reader what gets output.

And screen readers have a lot of control, so those are screenshots from VoiceOver on a Mac because that's available to me, and it makes sense to show it with what is available to me. But all browsers have a different set of settings. So this is the Verbosity setting that you can do. So you can say, with the button, I want the screen reader to output the name, status, and type. With the doc item, I only want the name and status and an image. I want the name of the image, the status, and the time, and so on.

So you can go in very granular and say which actual thing do I want. What verbosity? I can go in there and switch off all images and say, don't tell me the name of the status or the time of the image. I can do that if that makes sense. I don't know.

But if the user says images provide me with such little detail or information that I usually don't need to see them or even hear what their alternative text is, they can go in and do that. They can choose what they're comfortable with. And that's quite important

The same goes for punctuation. The user might opt to use most or all of the punctuation, only a couple of punctuations. So if you look at this repeated punctuation section, it says, First 3 Times. So if someone sends you an angry message and goes exclamation mark, exclamation mark, exclamation mark, exclamation mark, exclamation mark, exclamation mark, the screen reader is not doing what I was doing, which is very tiring and is not a lot of fun.

Read Numbers as Words is another thing they could also voice differently. When encountering a link or attachment, it could speak different things. So there's a lot of configuration there, and many users won't use it, won't configure it, but some will. You can even override certain abbreviations or symbols. You can even make VoiceOver pronounce "GIF" as "jiff" if you want to. I mean, you would be wrong, but you can do it.

The bottom line is there is no control in all of that, and also we have to deal with it. That's a GIF of three doctors putting on sunglasses, and the meme title says, "Deal with it." But there are more, other differences. Let's go really quickly through this figure element.

So figure elements-- they are amazing. We use them a lot because they make sense. Whenever you have an image or graphic or video, and you have descriptive text that you want to also have on the page, use the figure element. So let's go through this example of two birds, the South African swallow and the European swallow.

One is on the left, the other one is on the right. And on the left, the South African swallow has a different tail then the European swallow, which has this split tail with two long feathers, I guess. And the South African swallow has a short tail and lot of feathers.

It looks like a triangle, basically, while the European one looks more like a, I don't know, a fork. Just trying to paint a better picture here. But there are differences in support, obviously, for this new element that we have for 15 years now.

JAWS has the best support for announcing native figures and the captions, but the support is not perfect or consistent, depending on the browsers and JAWS verbosity settings. There are verbosity settings again.

IE11, if you wanted to announce properly, needs a role figure and aria-label or labelled by the points of the figcaption to mimic the native announcements. Edge, as the old HTML version, does not announce the presence of a figure or roll at all, even if you use ARIA, so that's out of the question.

Chrome and Firefox offer similar support, but JAWS and Chrome will completely ignore a figure, including the content of the figcaption, if the image has an empty alt text or is lacking an alt attribute. So if you don't describe your image, the whole figure is treated as presentation and removed from the accessibility tree. We don't think that makes a lot of sense either.

Testing NVDA, version 2018 something, no trace of figures. So a lot of different ways that figures just did not translate into actual real accessibility support, despite promising that that could work or would work. That's an issue. So this is probably the simplest thing to do for a figure is having the figure element with the roll figure and then ARIA label caption for the figure and then the figcaption. And if you add an image to the figure content, you have to add alternative text.

That's a lot of work for having an image and some text underneath it, and I don't think it should be like that. This stuff should be much more straightforward so that developers have a more easy way to do it. One thing is also mobile screen readers won't announce figures at all, but usually you can get into it and at least find the text and the alternative text and stuff like that. But it's not announced as a figure. Thanks, Scott O'Hara, for doing the research on that.

Yeah, it's just really messy. And even if you put ARIA on it, it does not mean that it just works for screen reader users because the support is so different, so wildly different. So ARIA can help but not in every situation. And accessibility is not only about screen reader users, which ARIA is made for. ARIA is made of screen reader technology, and that generalized and chose.

Most non-screen reader assistive technology does not work well with ARIA at all. One example is Dragon Natural Speaking. And this is the latest I found. It's from a 2014 article by Simply Accessible. Here is new support for a role button, role link, role radio, role check box, and ARIA label, which is great and which is a step in the right direction, but it does not cover the whole thing of ARIA. Click the menu. If Dragon doesn't know what a menu is, you can't do that, and stuff like that.

So close the dialog. What is a dialog? I don't have any idea, Dragon. Huge problem. This was the latest I could find. I hope the support has improved, but I'm not really-- I don't know, and I'm not really too positive about that.

But the other thing is Windows high contrast mode. On this screenshot, courtesy of Eric Bailey, those are all links on the left. You got two real links. You got a place holder link without an href. You got an ARIA link made out a div and an ARIA link made out of a button. And on the right, there's a paragraph of text with other ARIA links in it-- one ARIA link actually-- and without an underline.

If you switch on the Windows high contrast mode, the following thing happens. Real links are colored in yellow because that's the user preference there. Non-real links are not getting picked up and are not colored and yellow. So while on the left they are all underlined but not yellow, it's just weird for use of high contrast mode.

Why is this attractive thing not in the color that I specified? In the right paragraph, you only have a difference of color or hue. It's even not highlighted at all. And that's bad because people are unable to find your links, and that's always a red flag.

And the same goes for buttons. If you have a normal button, that shows up normally if you have a border around it. If you have an aria-disabled button, that gets no treatment at all because the Windows high contrast mode does not care about the aria-disabled being set or not because it only thinks that's for storage users.

If you have actually a button with a disabled attribute, it colors it green. It makes it less obvious than the normal button. So yeah, you can't rely on ARIA fixing different things, especially if they're not on the screen in a row.

I want to come back to users and how they are actually using our things, our application. Their knowledge level, obviously, varies greatly. There are users who don't know what their computers could do for them if they just pushed the right buttons.

Others hack their source of technology to work with inaccessible sites and just bypass inaccessible parts. Some users write style sheets and JavaScripts to change websites completely so they fit their needs. It's all over the place.

And users, they usually fit into this diagram, which is the type of disability, the proficiency with their accessibility tools, and the accessibility support. Users can be everywhere on the proficiency and type of disability scale, so they need different support. And then the accessibility support for your website comes in. And there's the third axis.

And your users can be everywhere. There are super proficient users who don't use the right accessibility tools for their use but still want to use your website. There are highly proficient users who use assistive technology that super aligns with their disability but has no accessibility support at all on your website.

And we have to square the circle here. Although because this is three dimensional, theoretically, we probably have to cube the sphere of the cube. Is that a thing? I don't know. Probably it's sphere-ing the cubes.

Yeah, so that's what we need to do, which is much more complicated, obviously. So lot of just so much of it is really complicated, and you don't know what happened. So I have to test with a diverse set of users, and most important you have to give those user agency. Trust that they use necessary technology that they know how to use.

Trust that the assistive technology suits their needs and allow for those assistive technology. Don't prescribe anything to them. Don't say, oh, you have to use my website in this and this way with this and this screen reader combination because that's not how it works.

Use HTML and CSS to describe elements to say, oh, this, the functionality of this button is this. The input element-- this is for name input. But I don't care what type of assistive technology you use, how you get the data in there, just describe it.

And we've seen that with password elements. We've seen that with password elements. People sometimes just disallow pasting in passwords into a password input field, and that's a huge problem for people with cognitive disabilities, people with memory loss. Don't do that. Don't prescribe how users need to use your website because that is bad.

I don't expect any proficiency from users. They come to your website, and it looks different than any other website they've been on because that's the power and some of the weaknesses of the web. The users come to your site, and it's specifically designed for your purpose.

So nothing looks like on other sites. And that can be disorienting. And so you can't just say, oh, the user will just figure out how to use my menu. We'll just figure out how to use it. You can't expect that.

And that's why conventions are so important. Don't break them for the sake of breaking conventions. That's not good. That's not what we want to do. We want users to be able to rely on conventions, and we want them to rely on to their settings and their preferences. And users get a lot of more tools to use.

Something that has become built in into many screen readers in the last few years is reader mode, which is great. Those are screenshots from the previous version of Safari's reader mode on iPhones. So you have the normal website, and then you can go into reader mode, and you have a black and white version.

You have a sepia tone background, you have a white on gray version, and then you have darker ones like white on black version. And you can also set the fonts, and you can basically customize that view to your liking. And there's like much more in there.

I love using reader mode. It's like one of the things where, you know, I'm confused. The website looks really like I need to scroll a lot to get to the content. I try reader mode. Usually, it brings the content of the website, boom, right in the middle. It cuts away all the navigation stuff that I'm not interested in when I want to read something, cuts away all the branding stuff that nobody cares about, and just gives the reader mode.

Safari has reader mode since 2010. That's 10 years of support. And you can set it to default for websites since 2015. So if you have a website that you get to regularly and you don't like their style, you can say, oh, show me this site always in reader more if it's available.

Firefox and Edge have reader mode since 2015. Chrome-- it's pending. It's supposed to be coming. Nobody knows anything. It has been in development, but it hasn't come far.

This reader mode in Firefox-- same principle, different styles, but you also have a reading mode. So it can read the content to you, which is really nice. So you can basically read with the content on the page. That's important for a user with cognitive disabilities, for example. And again, you can change the kind of style, you can change line length, et cetera, et cetera.

And there are even more website settings that get much more important all the time. So for example, you can set defaults. You can set never to autoplay videos. So for example, here on Medium.com, I say use reader view when available, so I never have to see the pop-ups and the stuff that comes with using Medium.

Then you can go into Settings and really go into details and say, never autoplay video here, but just stop media with sound on this site. That's on Safari, really granular, really good for users of that browser that they can set that. And then it just works. And the designers and the developers don't have to put in more work to make that happen. And I like that.

And then there are operating system level settings that you can use on the web as well. For example, there is invert colors, reduce motion, create grayscale, increase contrast, and reduce transparency that you can set in macOS and some of them also in iOS. So that means that in any native application that you go in and I say reduce motion, I won't have excess of animations, which is great, which is exactly what I want.

And we can use those settings in the web as well. So many of you probably have heard of the prefers-reduced-motion media query. It can basically say if prefers-reduced-motion is set to reduce, then no animation should happen on my page. By having an @media query, prefers-reduced-motion: reduce, and then you have star, a selector, and transition non-important and animation none important. You can do that. Support is quite good.

On different browsers, you can always look at CanIUse.com to verify that. On Windows, it's a little bit more complicated, which is why my good friend, Patrick Lauke, has written this short note on prefers-reduced-motion and puzzled Windows users, which is really good. And he also advocates for something that I also support, and that's defensive use of prefers-reduced-motion.

So instead of saying if the user prefers to have reduced motion, I reduce my animation. You do the thing that is the opposite. You say if the user has no preference for reduced motion, I will add the animation. And that basically means that when the user is using an old browser or browser that does not support prefers-reduced-motion yet, then you don't get any information. And I was a little bit thrown by that typo that I copy and pasted from Patrick's website. "Preferes."

The same goes with prefers-color-scheme media queries. This is called dark mode media queries. So you can go and you have your color set, and then you say, oh, if the user prefers the dark color scheme, I reset those colors to be the inverse or negative, or I have a vast set of colors that I put in there.

We will also get a lot of more tools to use-- and with we, I think designers and developers-- and they are supposed to come in Media Queries Level 5, but you know, stuff changes. Things get pushed and pulled into different levels at some point. Let's hope we get them.

Inverted colors, none and inverted. So you can say if the user prefers inverted colors or if the colors are inverted, that's actually not a preference because the user can just say do this. Then you can detect that, and you can basically undo certain reversions. That's useful if you have, for example, a design library, and you want to show different colors for your design.

That's the best example I came up with. But there might be other things that you want to do like having clearer colors or clearer borders around certain elements that just don't come out as much when the colors are inverted.

Then there's prefers-reduced-transparency, which basically means you can reduce the transparency of your stuff on the side. So if you have, for example, a transparent area on your page that is on top of a slider, you can make that fully opaque so that users who prefer reduced transparency don't get to see that transparency.

Prefers-contrast, high and low. So we have in WCAG 2 the 4.5 color contrast rule that sometimes it's only three to one. And now we can say, oh, people can indicate if they want high color or low contrasting colors.

So that means we start out with WCAG conforming 4.5 layout, and then we can say, oh, if someone prefers a high contrast view, we up it to seven to one, maybe even more. And if they like low contrast version, we can go and provide a low contrast version as well, which is useful.

And that might mean if you have low contrast between different things, you might also introduce different ways to separate elements, for example. There's a lot you can do if you have those media queries, and you can basically react to what the user is seeing and have some control over it.

And then there's forced-colors, which basically is Windows high contrast mode on steroids. So that means if forced-colors are active, you can react to that. One of the things would be like having thicker borders around your buttons just to make sure that they can be seen in any color combination, something like that.

And another thing that we are getting in Media Queries Level 5 is environment media queries. So that means how the environment is re-interacting with our website. For example, the light level. Is the light around my computer dim, is it normal, or is the content on my screen washed out because there's light shining on my screen? And reacting to that, I could increase the contrast, could make things nicer.

So those are the different ways that we will get control over what the user is using or how we can react to their preferences. And I think that will help us to make better, more adaptable websites for everyone. Thank you very much for listening.