Webinar: 10. BLH Training: Accessibility Fundamentals of WCAG 2.0/WCAG 2.1 Updates and WAI Resources

Date: 2/1/22

This is a Captioned transcript provided by CIDI to facilitate communication accessibility and is not a verbatim record of the session.

>> Johan Rempel: Good afternoon. Sorry about that. I was on mute. This is Johan Rempel from CIDI. And this week's presentation is accessibility fundamentals of WCAG updates and W3C‑WAI resources. I have the honor of co‑presenting with John Toles. Last week we talked about WCAG. Gave some examples. This week we will go deeper into WCAG and provide concrete examples. Including a demonstration of a screen reader as well.

So thank you to Heather for dropping the StreamText link into the chat. If you have attended any of the other trainings, we provide captioning for this. You can either access it through the link in the chat or access it through the closed captioning toolbar in Zoom.

This is being recorded. It will be archived and shared with Allie McDougall. Including the PowerPoint itself, saved as an accessible PDF. As well as the captions also.

And then a brief overview of Georgia Tech, CIDI, center for inclusive design and innovation. If you've attended others, you have heard a lot of this before because we are affiliated with a research institute, we do a lot of research in the area of digital accessibility and disability awareness. We provide a number of different service web accessibility evaluating, braille services, captioning and described audio, you heard Valerie Morrison speak about e‑text. Then we have various assistive technology specialists disseminating information, products and services related to assistive technology.

And John Toles will be presenting today. John has been employed with CIDI since 2016. He wears many hats, including customer support. He is one of our Drupal experts at CIDI. He and I work closely together related to a lot of digital accessibility initiatives that we're involved with in the ICT unit.

And y'all are familiar with me already if you have attended any of these trainings.

So the goals for today are to gain familiarity with the components necessary for a successful internet experience, why it's important to include people with disabilities, familiarity with strategies for writing, designing and developing for greater web accessibility and familiarity with the POUR principles and how they apply to web content.

So bear with me as I have the screen reader going with this presentation. I'm going to be providing a demo in a moment here. So I have this screen reader in my ear with every one of these slides.

So web accessibility: A unified experience with many components.

Keep this in mind that ‑‑ I'm sorry. Did I miss a question or comment? Maybe someone wasn't muted. So accessibility isn't just the responsibility of one individual. We all share responsibility when it comes to accessibility.

And there's several components to this. You have the actual content up loaded by content editors. Whether it's a Word document, PDF, PowerPoint, or even editing HTML on a page. It's really important to incorporate accessibility throughout that process. The CMS or the content management system whether Drupal or WordPress or whatever it may be. Taking that into consideration as well. The template being used and the level of accessibility. That has a huge impact on accessibility.

The browsers. If you're familiar with users having more success with one browser over another it's important to note that. If there are technical issues, it's important to encourage specific browsers. Very often chrome will work better than Firefox or vice versa with different content. So knowing that will provide additional information for people using assistive technology.

Keep in mind that not all media players are accessible. You're familiar with YouTube. YouTube is a highly accessible media player. If you're using other media players, proprietary media players, you need to make sure they're accessible. If they're not, point out what's not accessible and how to remediate those or navigate those for a more accessible experience.

Assistive technologies. Be aware of some of the assistive technologies that are used on the site and how those may interact effectively or not so effectively with various content.

And then evaluation tools. We're going to go into this a little bit today. But you will hear from Rayianna and John Toles next week on a whole host of automated testing tools that will allow you to enhance the accessibility of digital content by knowing what's accessible and what's not.

So this is really significant: The benefits of including people with disabilities in your workflow. I can't emphasize this enough. There's always temptation to use automated testing tools which are effective on some levels. We do that when we evaluate but to have individuals with disabilities at the earliest stage possible evaluate for accessibility is really crucial.

It provides a better understanding of real‑world experiences, the implementation of more effective accessibility solutions can benefit everyone. You know, one example is if you have an individual who may be color blind or low vision providing you feedback, that will help anyone who might be in a sunny environment or using a device or access to the internet in a location where the lighting may produce some glare. So it's one concrete example of the value that people with disabilities can contribute to whether content is accessible or not.

It helps maximize return on investment or ROI. And this is going to put it in very candid terms here but we were heavily active with the CDC distribution and still are with COVID‑19 materials. You know, they save lives. Having content available in an accessibility form at so individuals can be more well‑informed on how to protect themselves is one example. If you're operating as a business, you want to get the largest market share as possible. Considering that, depending on the metrics you use upwards of 20% of the population has some type of disability. This is a significant market share.

Reduce costly retrofits and redesigns. We have people interested in getting an evaluation of their website or application and they want everything in place and then evaluate it. That's not the way to go. A lot of accessibility issues can be handled at the beginning stages. Even at the mock up or wireframe stage to provide guidance on how to make that accessible before you invest that time and energy to build that out.

This is important: It increases motivation level to design with accessibility in mind. You have heard of personas, fictitious characters of individuals representing a certain demographic or profile. When you pull in people with disabilities that's very lucid in the mind of the developers of those who will be accessing content. So there's value in including people with disabilities.

A little bit about W3C‑WAI. It's the World Wide Web Consortium's Web Accessibility Initiative. It provides strategies, standards and resources to make the web accessible for people with disabilities. The initiative is responsible for creating the web content accessibility guidelines, also known as WCAG.

These are crucial when we talk about buy in and including people with disabilities. One of the questions we get is if we don't have necessarily individuals with disabilities available to test and be included in the evaluation and end product, what do we do?

One strategy would be to reach out to non‑profits. Maybe a blindness rehabilitation organization in your community. Potentially reaching out to individuals who are organizations of individuals who are deaf and hard of hearing for instance.

So there are resources available. This is another resource here is the perspective videos. They're very short videos. And we're going to include the URL at the end. They're 2‑3 minutes long but they do a good job of making accessibility issues come to life for the end user. So I encourage if there are people on your team who could benefit from this, definitely distribute these videos. And sometimes it creates an "aha" moment for individuals who don't understand how accessibility impacts the end user. And I will pass it on to John Toles here.

>> John Toles: Thank you, John. So, the next section we're going to cover is web accessibility: Writing, designing and developing.

So writing for accessibility: You want to provide informative and unique page titles. This is helpful for all users because it makes sure the title for the page or the element at the top of the page ‑‑ when you're looking at the top of the page is something informative so you know what the page is about. Especially if you're within a website and have a few tabs open as you're looking for particular information it will help keep you oriented even if you're a so-called user without disabilities. It's especially helpful for users using AT because that's how they track where they are within a set of webpages. You want to use headings to convey meaning and structure. So how you would lay out a Word document using headings, you want to make sure your webpage has headings as well. That's useful for all users because headings are what we scan for as we read through a page. AT users are able to navigate using headings. If they're looking for information ‑‑ say a friend referenced this website and they tell them the information they're looking for is under this particular heading. So they would open headings and listen for that particular heading.

You want to make link text meaningful. When you link to another page you want to make sure the user knows where they're going when they click on that link. That can be including context for the text around it but it's a best practice to make sure you're link text describes the destination that the user can expect to go to and that often includes whether or not it will open a new page. If you're linking externally and your habit is for any page linked to open a new window you want to write that.

You want to write meaningful text for alternative for images. You want some alternative text for images. You want to make sure your alt text includes a description of an image. If it's particularly informative for the content on the page you may want to include why it's on the page and why it's important. Some images are decorative. Those don't always have to have alt text.

You want to create transcripts and captions for multimedia. If you're creating a talking head video with maybe some graphics in the back ground you want that video to have captions. If you want to want to make sure it's completely accessible you will provide a transcript so someone can read through the transcript as well.

You want to provide clear instructions. Say you have a form included on the page, you want to include clear instructions for how to complete the form or if there's a task, you're asking users to do on the page, provide clear instructions for how to complete the task. And then you want to keep your content clear and concise.

There's a trend ‑‑ not really a trend. It's just human nature. When you're writing you tend to be overly complicated. You want to prove your worth with your writing degree, but that doesn't help all users. Most people on the internet when reading through content they read at an 8th grade level. That doesn't mean they're not capable of reading beyond 8th grade but if it's more complicated, they start scanning for key words. If you want users to engage with your content, you want to make sure your word choice and sentence structure is around an 8th grade level.

So designing for web accessibility: You want to provide sufficient contrast between foreground and background. We'll get more into that when we talk about WCAG specifically. That makes sure that you want to have as much contrast between the foreground and background.

You want don't want to use color to convey information. If you have required fields on the form the label text is red and that's the only indication for which fields are required, that's what we refer to as using color to convey information. The information about which form fields are required is based on color and not an indicator like the word required.

You want to make sure that interactive elements are easy to identify. If there's a button on the page you want it to have high contrast against the background. That it looks like a button. You want to provide clear and consistent navigation options. I think we're all kind of used to the way the web looks now where you have your menu at the top and then the page content that changes underneath the menu. Keeping it the same on every page is an example of providing consistent navigation.

You want to ensure that form elements include clear associated labels. So if you have a form, you don't want to just include the text inputs without having an associated HTML label because a screen reader reading through that form won't associate the ‑‑ automatically identify text as a label. It has to be coded as a label.

You want to provide easily identifiable feedback option. If you want to include a contact page. Give users a way to communicate with you and provide feedback on the site. So if they're reporting accessibility issues they know where to report to and there's back and forth communication between you and the user so they know it gets resolved. You want to use heading and spacing to group related content.

You want to make sure you have headings in your content. Then if you're including a lot of headings on a page you want to make sure they're spaced out enough so all the content isn't bleeding together but there's enough room between each section that it's identifiable as a section as you scroll through.

You want to create designs for different view port sizes. So that means most of ‑‑ 50% of all internet traffic now is on mobile devices. So you want to make sure your website is useable on a mobile device, as well as a PC or lab top or a desk top and a lap top.

And as many sizes in between on that. Tablet. You might have different view port arrangements for each one. For the phone everything will be more linear. So users will scroll through and have the same access to all options without having to span and zoom with their phone.

You want to include image and media alternatives in your design. So that means having alt text tags in ‑‑ alt text attributes in your image tags. It can also mean if you have a complicated image that's not describable in a short text that you would include in an alt tag then you have some sort of text alternative that describes that image. If you have a complicated infographic, you will have a short paragraph describing the image.

You want to provide ‑‑ sorry, John, there was one more left on that one. You want to provide controls for content that starts automatically.

Most pages don't include automatic background music or things like that but there is a design trend from a few years ago that pages have automatic carousels that scroll through.

You want them to be able to pause it and go back and forth. Not everybody is able to keep up with the speed of things that change automatically on a page.

The last section is developing for web accessibility. We talked about the labels on forms need to be coded as labels. You want to include alt text for images. If an image doesn't have an easily describe alternative then you want to code that correctly. You don't want a user coming to an image and hearing a description and then finding the secondary description of the image. That's okay if the paragraph is providing additional information to what's available in the alternative attribute for the image, but if not then you want to make sure the image alternative attribute is coded correctly so the image gets ignored.

You want to identify a page language and when the language on a page changes.

So usually in your base level HTML tag on your page you want to make sure there's a declaration of the language. If it's primarily English, put that tag there.

Any time the language changes, you want to make sure that there's a declaration. That's helpful for individuals who use screen readers. The annunciation will change. They will hear Spanish words with English pronunciation if not.

You want to make sure markup correctly. You want to make sure your buttons are always buttons. Your menus are always menus. Lists are always lists and marked up correctly. So that structural information to a user using a screen reader to interact with your content.

So you want to help users avoid incorrect mistakes. You provide instructions for the form if you ask your users to fill out a form. You provide instructions for the form. You clearly mark which fields are required if a required field is missing when a user submits the form. You want to make sure you point out which field is missing. If there's multiple form elements ‑‑ you want to indicate which one is missing or ideally all that are missing are indicated.

You want to reflect the reading order and the code order. The document object model, the DOM of the page should follow the natural reading order of the content. If the sidebar on the page is ‑‑ say you have a page with a sidebar menu, make sure that occurs after the main content. That makes sure you're not jumping back and forth between two different elements as a user is tabbing through the page because they have a mobility issue and they can't use a mouse and they use their keyboard but also for screen reader users. That element will have the same sort of navigation strategy as well by tabbing through elements on the page. You want to write code that adapts to the user's technology.

You want to make sure you're using the most current version of a coding technique so that the user has maximum compatibility with AT. The more you learn about how to code for including things we'll talk about later ‑‑ like ARIA and certain other things that you can do to make sure your code is compatible with assistive technology. The more you keep up with that, the more you ensure your code interacts well can assistive technology. You want to provide meaning for non‑standard interactive elements. That's talking about including specialized attributes on an element. Especially if it's not a semantic HTML element. There's certain things that you can include that will make sure that when a user encounters with their assistive technology the element is properly described via their assistive technology.

You want to ensure that all interactive elements are keyboard accessible. There's a lot of different coding methods to do that. If you can't get to an element with a keyboard, you're cutting out a lot of users. The keyboard is the most common piece of assistive technology. It's just having a keyboard and being able to interact with everything on a webpage.

You want to avoid captcha where possible. There are other coding methods that you can use. They're making strides all the time to make KAPTCHA more accessible but it's very hard to use CAPTCHA or accessible.

I think this is where you take over Johan.

>> Johan Rempel: Thank you, John. I'm sorry for advancing the slide a little early on you. So now I'm going to ‑‑ fingers crossed ‑‑ give you a demonstration of the W3C‑WAI before and after site. I have NVDA running, that stands for nonvisual desktop access. This is a common screen reader. The most popular one is JAWS. That's also the most expensive one. NVDA is free and in a follow up webinar training I'm going to be talking more about how to access both of these screen readers. I'm going to in real time now give a demonstration of the before and after site here created by the W3C‑WAI.

This is the before page. If you look at it ‑‑ with accessibility you can't eye ball it and say this is accessible and this isn't. You need the assistive technology. The automated testing tools will certainly lift a lot of this out and reflect on how accessible or inaccessible it is. The AT that we use for testing really is the true litmus test here.

This is the accessible version. If you are able to see the screen you probably didn't see a huge difference.

Let me load the website now and fingers crossed that this works nicely.

John, are you seeing the website displayed now?

>> John Toles: Yes.

>> Johan Rempel: Great. And you're hearing NVDA?

>> John Toles: Yes.

>> Johan Rempel: Good. All systems go. This is the inaccessible page. This is just a really good demonstration before and after page because you're not throwing any company or organization under the bus, right? Truthfully, it's hard to have any website 100% accessible at all times. The moment you up load an image that doesn't have a correct alt text or the moment you have content that the heading structure isn't correct the site becomes partially inaccessible. This is about a good faith effort to do the best you can related to accessible websites and the content up loaded.

Let's talk about headings. Headings are one of the most important ways that a screen reader user can quickly navigate through a page. You would think this would be coded as a heading "welcome to citylights" but in this inaccessible page ‑‑ which is very, very common ‑‑ these are not coded as headings. They stylized it to make it larger and change the color. So if I pull up the headings here ‑‑ this only has ‑‑ this page has one heading ‑‑ the "before and after demonstration." If I go to the accessible webpage, you'll see a huge difference here.

If I pull up the headings here, you'll see they're nested within one another here. You have the welcome to city lights.

>> John Toles: It's naught sharing the dialogue box.

>> Johan Rempel: No problem. What I can demonstrate here is go from heading to heading on the accessible page. Are you seeing the red highlight on the page, John?

>> John Toles: No. Just hearing NVDA saying it.

>> Johan Rempel: Can I skip to navigation menu, welcome to citylights heading level 2. [Synthesized speech].

Heat wave link to temperatures H2. [Synthesized speech].

Elsewhere on the web.

[Synthesized speech].

Elsewhere on the web. So you can quickly and easily skip through the headings on this page simply by pressing the H key or using the dialogue box that I brought up that you weren't able to see.

Let's go back to the inaccessible page. If I navigate through this the only heading is the "before and after demonstration." So it turns the page into a needle and a hay stack concept. You're also dealing with ‑‑ which I've been dealing with because I've been hearing NVDA in my head set and hearing people coming and leaving in the room and trying to hear the other feedback. It's auditory overload. There's a cognitive overload aspect as well.

Let's talk about the graphics on this page. Let's start at the top. I'm going to navigate through the graphics. [Synthesized speech].

We have the W3C logographic. I'm pressing the G to go through graphics. Those are labeled correctly. Listen to the next one. [Synthesized speech].

You can imagine how time consuming for someone who is blind to listen to all of that. I've worked with Georgia Tech students for years on the needs of individuals with disabilities and there seems to be a common trend of over explaining things. Someone who's blind generally doesn't want to know about all of those details. Again, with that cognitive over load and the auditory fatigue that takes place it becomes overwhelming.

Let's go to the accessible page now. I'm going to skip to the same graphic.

[Synthesized speech].

Citylights your access to the city. Graphic.

>> Johan Rempel: That's it. We don't need to know there's a white C inside a red circle and all that. Try not to over explain graphics. Unless it's absolutely critical to what you're accessing. Let's go back to the inaccessible page.

If I navigate through. [Synthesized speech]. Graphic link. So there's real value in using the null attribute which actually skips graphics. These graphics don't describe or provide information other than what's already there with the home text and news and tickets and survey text. It simply is auditory overload to describe it that way.

So, creating those null attributes for those additional images is going to streamline the site a bit.

[Synthesized speech]. Be cognizant of this. This bullet graphic was highlighted. The killer bees. It doesn't need its own graphic. It's additional information to tab through but doesn't serve a great purpose there.

So let's talk about links. John talked about the importance of contextual links. What I can do is go through the links here. These kinds of links like "click here." For some reason it's not.

[Synthesized speech]. There's a "click here." It's not selecting it here. These kind of links "click here" "read more, "they are noncontextual. They are also duplicates. So you have two click here links. It would be impossible unless a screen reader user navigates around that link to determine what that might refer to. So a big issue there with noncontextual links.

Let's talk about forms. We have the quick menu form. I'm going to go to that now.

[Synthesized speech].

Okay. Combo box collapsed. If I navigate down in this it would automatically launch me into another location. I will go to the accessible page now. If I go to the form here ‑‑

[Synthesized speech].

So this operates well. I can expand it and collapse it. When I'm ready to choose, I can tab to the go button. What you don't want to do is have triggers that simply by navigating through it that it launches into another location or activates something because the tab key and the arrow keys are really the same way ‑‑ they're used to explore the page by someone who's blind. It's similar to you looking at a page. If that exploration triggers events, then that's going to really confuse an individual using a screen reader.

So those are brief strategies for how someone who is blind accesses a site. This is a very basic site. When you're dealing with a site with hundreds of links and graphics this streamlining it will make it a whole lot accessible and useable by someone who is blind or visually impaired or maybe using the keyboard interface on some level either with a switch device or other device. Now I'm going to pass it back to John in just a moment. All you, John.

>> John Toles: Thank you. Now we're going to go through detail about WCAG and real‑world examples of elements and things that either meet or fail WCAG. Next slide.

So, WCAG is a web content accessibility guidelines. It is web content that includes content on the webpage and text and multimedia and code and mark up that creates the webpage. So it's everything about the webpage. So the World Wide Web Consortium's web accessibility initiative developed these standards. The latest version is 2.1. You'll see a lot if you keep up with the current legal landscape. You'll hear 2.0 referenced a lot. It has since been updated to 2.1. 2.1 doesn't supersede 2.0. It includes all of the guidelines from 2.0 and additional criteria.

So the guidelines are broken down into four categories: Perceivable, operable, understandable, robust, the POUR principles. On this slide there's a link to the W3C‑WAI homepage and the WCAG 2.1 quick guide.

Next slide.

So the first category of guidelines is perceivable. Your content should be perceivable by people using a variety of strategies, including visual, auditory and even tactile. A person who is both blind and deaf can still use the internet with an assistive device called a refreshable braille display. On the right-hand side you see a person using a refreshable braille display. It displays text on the page on the braille display.

To describe the four sections means it's important non‑text content has text‑base alternatives. Complicated images need to have description. Audio and media need to have alternatives for people who cannot see or hear them. If you're hosting a podcast and you want a transcript of the podcast if you have video media on the page, you want to provide a transcript for that as well. Or you could provide an alternative audio track on the video that includes audio description.

Content is presented in different ways when appropriate without losing overall context. That means you can change the layout of the page and not lose the overall content. So if you go from reading on your desk top and then pull up the same information on your phone. The information doesn't change. It's just a different presentation for the end user. You want to make sure your content is easy to read, watch, listen to.

So some perceivable guidelines. The most important one is 1.1.1 level A. So the success criteria are divided into three different levels. A, AA and AAA. Level A and AA are basic standards and AAA is going beyond and beyond into best practice.

So back to the glide, level 1.1.1 is level A. It's very important. All non‑text content that is presented to a user has a text alternative that serves the equivalent purpose. There's some exceptions for things like ‑‑ that are outlined in the WCAG guidelines themselves. So exceptions for mainly logos. You don't have to provide a complicated description of your logo. You can just say this is my logo. You don't have to make sure that the user knows what it looks like. All images that are important to the content on the webpage must have text alternatives. Developers and content editors can include text on an image by using an alt attribute. There's an image of some golden retriever puppies running in a field. There's an alt attribute that says a litter of golden retriever puppies playing in the grass. A good alternative is short and concise and avoids using words like image or graphic. Including that is redundant because the screen reader will announce that.

We talked about this a little bit earlier but if an image was chosen to convey a particular mood the alt text is a way to convey that as well. Good alt text can help users who are experiencing issues with their computers or limited bandwidth. The alt text will replace the image with a fallback when the image doesn't load properly. So there's an example on the right-hand side. The image doesn't load so the alt text is displayed there.

Next slide.

Another important guideline under perceivable is 1.4.6. That's level AA. It's contrast minimum. The visual representation of text and images of text has a contrast ratio 4.5 to 1. So the contrast ratio is how the W3C has decided to measure the contrasts. They take the foreground and background and run it through an algorithm. There's others that we'll get into in a little bit. Meeting the contrast minimum guidelines means that the color of the text on the page is different enough from the background color to make it visual for users. So 4.5:1 describes a ratio between the text of the background meaning one is 4.5 times lighter or darker than the other. There's links to tools to test color contrast. The color contrast analyzer from Paciello group and then the contrast checker from WebAIM. There's an example on the right of pink text on a purple background which is difficult to read. There's some logos for the color contrast analyser and the WebAIM checker.

So larger text is a font size of 18 point. That's point not pixels. 18 point is equivalent to 24 pixels. So the most stringent standard for contrast ratio for text is 7 to 1 for level AAA but it's very hard to meet that standard for anything other than black and white. WCAG also includes contrast guidelines for nontext elements like buttons, clickable image links, custom page controls and the ratio for those is 3 to 1 as well.

The next category is operable. Users should be able to operate any navigation or control through a variety of means. The most common assistive device is the keyboard. Many devices use keyboard input standards. If you make your content accessible using the keyboard, you will make it accessible with a lot of other assistive technology as well. Things like navigation switches and ‑‑ things like that.

So, guidelines in this category are subdivided into 5 sections: Keyboard, accessible, enough time, seizures/physical reactions, navigable and input modalities.

Page content and controls should operate via the keyboard as well the mouse. You don't want to exclude one for the other. Users are allowed enough time to read the content, complete any inputs being requests and content doesn't change automatically. Any visual effects or animations included in the content meet thresholds known not to induce seizers and navigate your webpage with the keyboard.

Some important guidelines for operable 2.1.1 which is level A. That's the keyboard. All function alt of the content is operable through the keyboard interface without requiring timing. There are exceptions to this guideline such as handwriting input using a stylist. In these cases that doesn't mean you can't ‑‑ you don't have to make it keyboard accessible. You have to provide an alternative that's keyboard accessible. Keyboard users navigate through webpages using the tab key. A common situation for keyboard users is to find interface components that don't receive keyboard focus when tabbing through the page or don't activate via the enter key. Developers can avoid these situations by preferring semantic HTML elements. On the right-hand side is a zoom text keyboard which has large text on the keyboard keys and has at the top various short cut buttons have working with Zoom text which is a magnifier software. And there's an example of a properly coded button here ‑‑ custom coded button. There's a button that says click me. It's been provided role button and given tab index of zero so it falls within the base level tab order on the page. Even though that button is accessible, it would be better coded as button tags.

Another important level is operable 2.4.7 level AA focus visible. Any keyboard operable user interface has a mode of operation where the keyboard focus indicator is visible. Not all users who rely on the keyboard device because they cannot see. Many keyboard users typically don't use screen reader when would announce the content. Developers and designers sometimes disable the native focus on a browser for aesthetic reasons but also forget to take into consideration focus when creating custom controls. There's an image of a navigation menu that is using the base level focus indicator which is a black dotted square around the element.

It's best to enhance when available and there's a focus selector so it puts a two pixel around the element. There's an image of a solid blue boarder around it indicating it has focus.

Next slide.

Are we going to 4:15?

>> Johan Rempel: Correct. 4:15.

>> John Toles: I will try to get through this next section faster so we have time for questions at the end. Next section is understandable. Users must be able to understand the content and how to use the controls on your webpage. Guidelines in this category are subdivided into 3 sections: Readable, predictable and input assistance. That means the language of the page is identifiable programmatically for screen readers, mechanisms for designing unusual words and abbreviations... [Reading]. There's an image of some grade schoolers using a computer here to illustrate the fact that if your content is ‑‑ if your audience for your content is grade schoolers you won't write the same way for high schoolers or the general public. That's to help them understand the content.

Another is understandable 3.1.1 level A language of the page. The default human language of each page can we programmatically determined. Screen reader respond to the language identified on the page by changing their pronunciation. The easiest way to set the language for screen reader is using the lang attribute. There's an example there of an HTML tag with a lang of English.

Next slide.

Another guideline would be for understand is 3.3.2 which is level A. Labels or instructions. Labels or instructions are provided when content requires user input. Develops and designers often expect the purpose of the inputs to be obvious... [Reading]. There's an example on the right-hand side. You often find this is just being the magnifier but in this instance it's the word search and the magnifier. This will be accessible for a user that's listening to it with a screen reader.

And the final category we're going to talk about is robust. Your content can be interpreted by a wide variety of current technologies and assistive devices and it will be compatible... [Reading].

Next slide.

So, I think this is the only guideline we have in this section but it's 4.1.1 level A parsing. In content implemented using markup languages, elements have complete start and end tags, elements are nested according to their specifications, elements do not contain duplicate attributes or IDs are unique. Most automated accessibility testing tools can scan for these issues. Screen readers and other assistive technology interact with content on the code level. And there's 3 links on this page for some automated testing tools. Wave from WebAIM, axe for Deque.

There's another robust one. The other important guideline for this is 4.1.3 which is level AA stratus messages in content implemented using markup languages. Status messages can be programmatically determined through role or properties such as that they can be presented by the assistive technology without receiving focus. If a user input something into a /TPAORPL, the user must be alerted about the error. Manual testing must be done. There's an example of a dialogue box "thank you for subscribing" and the instructions for how to unsubscribe and there's also an image of a Java script alert on the right-hand side as well.

Next slide.

So some of the other ‑‑ I think we're almost out of time. These are some other standards. The authoring tools accessibility guidelines and the user agent accessibility guidelines. They're similar to WCAG but the authoring tools are specific for tools that you use to create content. And the user agent accessibility guidelines are related to browsers.

Next slide.

And then this slide is about WAI ARIA. That's what I was talking about earlier. WAI‑ARIA is how you describe custom content and controls to a user's assistive technology. There's a link to instructions to the introduction to WAI‑ARIA. If you're a developer and you have a lot of custom controls on your page, I suggest you check that out because it's extremely important for making sure that all of your controls are accessible.

Next slide.

Here's additional resources. These are links to some of the earlier content, the perspective videos, the writing for web accessibility guides on the W3C webpage and the before and after website and the introduction to WAI‑ARIA and the automatic testing tools. I think we're coming in just under the wire. If anyone has questions you can post them in chat or feel free to unmute yourself.

>> Johan Rempel: I know you had a lot of material to go through so you're probably enjoying a talking break now. John put a lot of work and energy into these slides. Any questions? I know it's a lot of information. If you're not familiar with some of this, it probably feels like drinking from a fire hose. Any questions from anybody? Keep in mind you'll also have access to this PowerPoint as well to review. John has provided some really good resources. I'm going to the chat. There's a question. I really appreciated hearing JAWS do its thing. That was actually NVDA. I know that was a comment from previous. That was NVDA. The reason I didn't run JAWS is JAWS is a little bit more memory intensive and less versatile than NVDA. It takes up a lot more system resources. I didn't want it to crash. I'm glad it was helpful. It would have been helpful to see the dialogue boxes coming up and the programmatic dialogue boxes taking place. With the back and forth I couldn't do it. I could share the application but not the screen this time.

Any other questions or comments?

Next week John Toles and Rayianna Daniels will cover automated testing tools. All of the tools will be covering are free. So if any of you want to start tinkering around with them and learning them, next week's session will be a good opportunity to learn more about the pros and cons of those.

All right. If no one has any questions we will close it out here. Thank you. I know you're all busy. Thank you for devoting time for this. We appreciate it. John, any closing comments before we close it out?

>> John Toles: No.

>> Johan Rempel: All right. Thank you for co‑presenting with me again.

>> John Toles: You're welcome.

>> Johan Rempel: With that we will close it out. Thank you so