Accessible Audio Transcript for "WCAG 2.0 top ten checkpoints: Ten popular areas of focus for compliance and accessibility" Video

[Intro page slide content: Brought to you by The Commission of Deaf, DeafBlind and Hard of Hearing Minnesotans and The Office of Enterprise Technology: State of MN]

Welcome

[Instructor speaking] Hi. Welcome and thank you so much for coming to today's presentation on WCAG 2.0 top ten checkpoints. We're going to be talking about 10 popular areas of compliance and accessibility in regards to WCAG 2.0.

1. Forms

The first area we're going to talk about is with web forms. "Web forms" is a popular problem within state government in terms of making them compliant.

[Slide Bullet: Label tags for ALL input points]

The first area is to make sure that all points of input have a label tag.

[Slide Content on Screen:]

Example one:

<label>First name: <input type="text" name="firstname" id="firstname" /></label>

In this example code that I have got here, we have two different ways of making a point of input accessible. The first is to actually surround the point of input with label tags (example one as noted above).

Example two (recommended):

<label for="firstname">First name:</label>

<input type="text" name="firstname" id="firstname" />

And the second is to actually provide the label tag with a "for" attribute that actually connects it with the input tag. If you use this sample code, that point of input will be accessible (example two as noted above).

[Slide Bullet on Screen: Correct tab sequence]

The second area to make sure you have the proper tag sequence in your form. If there's one thing that frustrates end-users is when they're tabbing through a form and they're suddenly out of order and out of sync. Make sure that you test your forms to be able to tab through in correct sequence from input to input.

[Slide Bullet on Screen: Navigable and able to submit with keyboard]

The third area is to make sure it's navigable and submittable with the keyboard only. You should be able to navigate through each point of input with the keyboard only and not rely on mouse clicks to submit the form. This is an important area of concern because there are a number of users that don't have access to a mouse or rely upon keyboard only submission for navigation.

[Slide Bullet on Screen: Access keys for complex, long and laborious forms that are used frequently]

The fourth point is when you have forms or Web applications that are used frequently by your employees for entering large amounts of data you might want to consider access keys to access frequently used areas within the form. It's not widely supported but it's something to consider if you have laborious forms for people to enter repeatedly on a daily basis.

Oh, and you've got a question?

[Participant speaking] Yes, I was wondering if I should use JavaScript to validate forms?

[Instructor speaking] Good question. Form validation is okay using JavaScript but you don't want to rely on it solely. You should always make sure that you're validating server-side as well as client-side. Using JavaScript to validate your form as the user is typing in data is good to help guide the user while they're inputting data into the form. So good question. Thank you.

2. Document structure

[Instructor speaking] The next topic is document structure.

[Slide Bullet on Screen: Title tag to convey relationship of page to the site]

First point, you want to make sure you're using the title tag to convey the relationship of the page to the entire site.

Here is an example of a snippet of code.

[Slide Content on Screen:]

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" **lang="en">**

<head>

<title>Page Name / Site Name / State of Minnesota</title>

In this example, I'm showing you how to actually set up your title tag. We're going from the page name to the site name and in this example, going to the state of Minnesota as our entire context. It's from specific context to most generic context. So you have a full-breadth view of where you are at in relation to the site.

[Screenshot of a multi-tabbed application displaying two tabs: Images in Content and CSS-Controlled Layout]

I also have got a screenshot here (noted above) that shows an example how this looks in a multi-tabbed environment. Web browsers have gone the way of tabbed environments and when you have multiple tabs opened, sometimes it’s hard to get an at-a-glance view of which page you're looking at or which pages you have opened up in your browser. It's usually helpful to do that.

[Slide Bullet on Screen: Informative and meaningful heading tags]

Also, you want to make sure that your heading tags, which are the H1, H2, H3 etc., that you're using informative and meaningful text within those heading tags. Don't just make it “contact us.” Use your headings to make it actually meaningful for your users to get a picture of where they are and what the page is.

[Slide Bullet on Screen: Headings are navigable; use to convey structure]

Also as a point of note, headings are actually navigable for users using screen readers and other assistive technology. They're able to skip through the headings in your page. So if you have a long document in your webpage, users can actually skip through the different headings within the document. So it’s really useful, which is another reason to make it meaningful as to what the purpose of that piece of content is and where it belongs.

Yes, you’ve got a question?

[Participant speaking] Yes, is it enough for my headings to be larger and bold?

[Instructor speaking] That's a good question. Actually you shouldn't rely solely upon visual information to convey the difference between content and headings.

[Image of a word document displaying various headings and blocks of text]

Here is an example of a screenshot that I’ve got that illustrates the difference in the headings and how it conveys relationship with the information. We’re starting out with our heading one tag. Here it says produce. And then we have an introduction paragraph and we’ve got a little submenu. And it breaks down into a category of vegetables. That might be our heading two tag which would be a subcategory or a sub-topic of produce. And then within vegetables we’re breaking down into what would be heading threes. We've got our corn, carrots, peas and other vegetables. So we’re conveying relationship of the content in its overall structure. So that's why you don't want to rely solely upon making something bold or 15 points larger. In fact you should be using stylesheets solely to format your content, especially your headings.

[Slide Bullet on Screen: Do not rely on visuals to differentiate head]

So don't rely solely on visuals.

3. Navigation and links

[Instructor speaking] Alright. On to our next point. We’re going to be talking about navigation and links.

[Slide Bullet on Screen: Consistent navigation (predictable)]

First item, you want to make sure that you've got consistent navigation and that its predictable. For some static websites, it's a little bit harder to stay consistent because you may be using the same HTML on every page and if one page changes, your navigation might be out of sync. You want to make sure you’ve got consistent navigation and the set up and expectation is clear for all of your end-users.

You've got a question?

[Participant speaking] Is it alright to use menus with JavaScript functionality, like with the fly-out sub-menus?

[Instructor speaking] That's a good question. Actually it is okay to use JavaScript for some of that functionality and to have the fly-outs in your menu. But the thing is you don't want to rely upon it solely because there are some users that may have a degenerative condition in the hands that are unable to control the mouse. They might not have the fine motor skills to be able to control a mouse as easily. So you might want to consider providing a secondary level of navigation. For example, in this screenshot here, we've got an OET website. We do have that fly-out menu functionality where a user can navigate that way but we would also have a secondary level of navigation so the user can easily just tab through the navigation and not have to worry about, you know, the menu disappearing because they can't control the mouse as well. Very good question.

[Slide Bullet on Screen: Skip to content]

The next area we want to talk about is skipping to content. If you've got a long menu and screen readers, just as an aside, will go through and read all the navigation. So if you're using the unordered list to present your menu, it's going to sit there and read through all of the navigation. You want to make sure you provide a way to skip down to the content and it not only benefits users that may be using screen readers or Braille readers, but also for users that might have a mobile device that have the old scrolling trackpad and giving them the ability to skip past the navigation to the content is priceless.

[Slide Bullet: Navigating with anchor tags in long bodies of content]

The next area I want to talk about is navigating with anchor tags through long bodies of text. If you've got a long page of content, you might want to consider using anchor tags to jump down to different sections. Yes, users can navigate with headings with assistive technology such as JAWS or other screenwriters but for users that just use a normal browser without that sort of assistive technology, sometimes it's beneficial to give them the ability to skip down a section. And this also applies to mobile users. You're making their lives easier that way too.

[Slide Bullet on Screen: Breadcrumb navigation]

Alright. Here's an example of using breadcrumb navigation.

[Slide Content on Screen:]

Home > About > **Our Vision**

```
<div id="breadcrumb">
   <ul>
      <li><a href="index.html">Home</a></li>
      <li><a href="/about/index.html">About</a></li>
      <li><a href="/about/ourvision.html" class="current">Our Vision</a></li>
   </ul>
</div>
```

In the screenshot above, it starts with Home, About and Our Vision. Our Vision happens to be the page that you're on and it gives you an overview of where you are in relation to the entire site. You may not use this if you've got a small agency and you've got maybe 10, 15 or 20 pages. That's no big deal. If you're using consistent navigation, they will be able to find where they are but if you are a large organization and you've got hundreds and hundreds of pages, maybe even thousands, breadcrumb navigation is essential to give the user a snapshot of where they are in respect to the entire website.

[Participant speaking] Is it OK for me to use "Click Here" as my text link?

[Instructor speaking] Very good question. One reason would be for screen reader users that use assistive technology are able to get an at-a-glance view of all the links that are on a page. So let's say, for example, you have a page that has a list of articles and let's say that every single one of those links say "click here for more" or "read on." Imagine being read to you all these lists of links that say click here click here click here click here click here. It doesn't offer the user any sort of context as to what that link actually is. So rather what you want to do is provide meaningful link text.

[Slide Bullet on Screen: Meaningful link text that conveys purpose]

So let's say you've got a sentence. Always hyperlink the subject matter of that sentence, not the action. As we all know, we've got to click on a link. That's second nature but what you actually want to do is hyperlink the subject because as users, we've become accustomed to just skimming through content and if you actually hyperlink the subject matter, users are actually going to see at-a-glance what that link is all about and where it's taking them to. So not only does it benefit users that have some sort of visual impairment and use a screen reader or Braille reader, but it also benefits us all in terms of if you're a skimmer. You can see at-a-glance what that link is all about. Very good question.

4. Images and non-text elements

[Instructor speaking] We're going to jump to the next section about images and non-text elements.

[Slide Bullet on Screen: ALT tags for informative images (non-decorative)]

First thing is providing alt-text for informative information or in images that may be embedded in your page. So if you've got a graphic chart, you want to provide alt-text within that image if you're embedding it in the HTML.

[Slide Bullet: Link to descriptions for longer text block]

If you've got tons of information in that chart, for example, you may want to consider providing a link off to a text description or outline of what that information is all about. Sometimes it might become a bit much to actually put in an alt tag.

If you've got decorative images, however, you may want to consider actually using CSS to format those and not present it in the in-line text.

I have an example here of two images.

[Image #1: a man with the title “We’re here to serve you.”]

[Image #2: a man with the title “Our Service Model” along with the following steps: 1. We Get Your Call; 2. Figure Out the Problem; 3.We Solve It]

One is an image that is arguably decorative and the other is arguably informative. The first image shows this guy leaning on his hands and it says “we’re here to serve you.” The second one actually has some text in it that describes the service model process. One, we get your call. Two, we figure out the problem and three, we solve it. The image that is informative we might want to actually either embed in the content or least provide a link to the descriptive text about that content. If you provide the image in-line with your text, you actually want to make sure that you've got an alt tag available for that with the description in the image.

[Slide Bullet on Screen: Decorative images presented with CSS (not in content)]

Otherwise with the decorative image, make sure you’re using stylesheets or CSS to actually present that image so you’re not embedding any necessary images within your content. Okay?

[Slide Bullet on Screen: Contrast ratio between background and text]

The next point I want to say is that there has to be sufficient contrast ratio between the background and text. And this also applies to say audio and video to make sure there is sufficient contrast between the voice and maybe some background music or any ambient noise. But for images, WCAG set up a specific ratio for the contrast between text and the background. You can find more of that information provided later and links actually to WCAG.

[Participant speaking] Is it okay if I use flash on my website?

[Instructor speaking] Good question. You can actually use flash. There are ways to make it accessible but you want to make sure that you're following Adobe's guidelines. But all of the other principles apply. Don't rely upon visual stuff. Make sure it can be navigable by keyboard only and not require mouse clicks. But generally speaking, it's okay. You just have to take a few extra steps to ensure and test that it is indeed accessible.

5. Tables

[Instructor speaking] The next section we're going to talk about tables, one of my hot buttons.

[Slide Bullet on Screen: Do ***NOT*** use tables to format your document]

The first point is do not, I repeat do not use tables to format your web page. I confess. I've been there. I've done that back in 1997, 1998 or 2000 when CSS was still in its infancy. Yes, we used tables to format our web pages. But now, with the luxuries in the technology available to us through the cascading style sheets, do not, for the love of all things holy, do not use tables to format your page.

Yes, question?

[Participant speaking] Is it okay to use tables to control the layout of my web forms?

[Instructor speaking] Good question. I've known many a developer that takes a short quick shortcut to actually format their forms with tables and do not do that. The reason being is that screen readers read out the output from top to bottom as the code is rendered. If you've got tables that may be out of sync with the order that the form should be read out, you're going to have problems. Instead be familiar and comfortable with style sheets enough that you can actually format your form solely with CSS and actually keeping all of the formatting out of the code itself. The ultimate objective is you want your page content to only be about content and not about format. So use and rely upon style sheets and not tables for formatting. Very good question. Thank you.

[Slide Bullet on Screen: Tables are for tabular data]

Tables instead are meant for tabular data. So if you've got a table of figures for say financial figures for a range of years, yes, most certainly use tables for that sort of data.

[Slide Bullet: Use THEAD, TFOOT, and TBODY tags to convey data relationship]

In fact, go the extra mile and use the THEAD, the TFOOT, and the TBODY tags to actually convey the relationship of that data with each other. I will provide some example code here that you can look at the end of the presentation. But as I mentioned before, you want to make sure that in your table itself, when you're actually using it, you actually keep all the formatting out of the table itself and just use it for actually presenting the data.

[Slide Bullet on Screen: Rely on CSS for formatting and avoid depreciated tags]

Use CSS to actually handle the formatting and avoid depreciated tags.

6. Mouse and keyboard issues

[Instructor speaking] The sixth area that we're going to cover is about mouse and keyboard issues.

[Slide Bullet on Screen: Test and ensure you can navigate with keyboard only]

First of all, you want make sure you test and ensure that you can navigate to your page with the keyboard only and not requiring the intervention of your mouse.

[Slide Bullet on Screen: Do not rely upon mouse clicks]

Don't rely upon mouse clicks. Sometimes I've seen users that build forms that actually require a mouse click to submit the form or might require a mouse click to actually interact with the website. Be cognizant of that to make sure that the user is actually able to navigate and perform the functions of your website with the keyboard only.

[Slide Bullet on Screen: Be cognizant of tedious clicking issues (e.g. menus)]

Also be cognizant about tedious mouse clicking issues. We talked earlier about the JavaScript menu fly-out functionality and providing an alternative for users to go and navigate with the keyboard only. Actually, there is a wide set of users with varying circumstances that make it important to navigate only by the keyboard. For example, you might have a reporter who's got a repetitive stress injury and it's difficult for them to use the mouse but it's little bit more comfortable by navigating through the tab key. Or for example, someone who might have Parkinson's or essential tremors or they have a very difficult time controlling the mouse

movement. So with the tab key, it's consistently there and it’s a little bit easier for them to navigate through.

So persons with different sort of circumstances that make it essential to navigate only with the keyboard. And to some degree, it requires a little empathy and the ability to step outside of your own circumstances and think about what it's like to browse the web with your eyes shut and listening to a screenwriter or trying to navigate through with an archaic palm pilot or whatever. It's understanding the different circumstances that helps make it a little bit easier to implement some of these things.

7. Client-side to server-side handling

[Instructor speaking] We’re going to move onto the next section about client-side to server-side handling and this deals with JavaScript and jQuery.

[Slide Bullet on Screen: It is OK to use Javascript and jQuery]

It is okay, and I do want to dispel the myth that it's not okay, but it is okay to use JavaScript and jQuery to handle client-side functionality. But that said, there are certain measures that you need to take in order to make sure it's accessible and we’ll dive into that. Yes, do you have a question?

[Participant speaking] Yes, what about Ajax? Is making Ajax calls accessible?

[Slide Bullet: Build base-level, server-side functionality first]

[Instructor speaking] Yes, it is okay to use Ajax but at a fundamental level, you want to make sure that what you do is build your server-side functionality first.

[Slide Bullet on Screen: Add your jQuery and Javascript functionality on top]

On top of that, you add your jQuery in your JavaScript functionality on top. Because what you want to make sure is that your form or your application or your website or whatever you’re building, functions without JavaScript. So if you’re pulling in external data in an Ajax call, make sure that whatever link or button or the technology you’re interacting with, make sure that it goes and directs the user to that actual data and then brings them back.

So it's more complicated and there’s a whole lot of information out there available on how to make JavaScript and Ajax and jQuery accessible but at a fundamental level you want to make sure that you’ve got that server-side functionality first because that'll work in anything. It will work in a mobile device, it will work with assistive technology to work with current and older generation browsers.

[Slide Bullet on Screen: Have a plan to degrade from client- to server-side]

If you've got an application, however, that you get from a vendor or another client or another party, you want to make sure that they, as well as you, have a plan to degrade gracefully from client-side to server-side. It's always complicated when you got a vendor involved but at a fundamental level, you want to make sure that there is a plan for that application to fail gracefully if the user has JavaScript disabled or maybe they don't have the luxury of style sheets or what have you.

[Slide Bullet on Screen: Inform user of user input and changes in page behavior]

Also, you want to inform the user of input changes in the page behavior. That's probably the number one issue with making Ajax calls. Most assistive technologies will actually use browsers natively such as Internet Explorer and Firefox. For the Mac environment, voiceover client uses Safari natively. But you want to make sure that you notify somehow or instruct the user or give them cues that there are actual changes occurring within the document model. There have been great advances made within the ARIA in terms of Internet rich applications in terms of how the user is notified that something is changing in the DOM. But that's the number one issue, to make sure that you're informing your user of any changes that are happening in the page because of their input.

8. Cues, instructions and error handling

[Instructor speaking] Alright, let's move to the next section. Number 8: cues, instructions and error handling.

[Slide Bullet on Screen: Inform and instruct the user of your input expectations]

The first thing you want is to inform and instruct your user of your input expectations. For simple contact forms, you're probably not going to have to provide in-depth instructions.

[Slide Bullet on Screen: Provide help information and guide user input]

But if you've got a web application or you've got form input that you're requiring more detailed data or more specific data, make sure you're instructing the user of what your expectations are. In many ways, you can actually, to this next point, help the user and guide their input. You can actually use like JQUERY. We talked about JavaScript for handling and processing forms. You can actually use JQUERY to help guide and mask the user's input. Say you need a phone number inputted a very specific way. Provide help to actually guide the specific formatting and you can do that with client-side scripting, and you can do it with simple text for instructions as well.

[Slide Bullet on Screen: Offer intuitive error messages]

Now if the user doesn't input a form properly and they actually run into an error, what you want to do is you want to make sure that you're providing an intuitive error message that makes sense to them and clearly articulates what the problem is, where they went wrong and even giving them the option to jump right to that form field where the error occurred.

[Slide Bullet on Screen: Best of all make your forms intuitive]

But best of all, make your forms intuitive. It's a challenge especially with state government or any government agency. We like to make things more complicated and difficult. The bottom line is make sure whatever you're putting out there is intuitive and it's simple to use.

Yes, you have a question?

[Participant speaking] Isn't it enough that I just mark my required fields with an asterisk.

[Instructor speaking] Good question. Thanks for actually asking that. It's a good starting point. It at least guides the user in terms of what is expected of them to be inputted, but you also want to rely upon other textual information to give cues and instructions as to what is required and expected of them. Thanks.

9. Display adaptation

[Instructor speaking] Number 9: display adaptation.

[Slide Bullet on Screen: Support multiple browser environments]

First of all, you want to make sure that you provide support for multiple browser environments. This means not boxing yourself into "oh, we only support Internet Explorer." Make sure that you're providing support for other browsers such as Firefox, Safari and Google Chrome. By doing so, you actually broaden your user base because there are some users that don't have access to Internet Explorer or there are some users that are browsing with a mobile device and they don't have the luxury of choosing which browser they're using. So make sure you're providing support for multiple environments.

[Slide Bullet on Screen: Content is format-neutral, format only with CSS]

Also make sure that your content is format neutral and formatted only with CSS. This way your content is extensible and you can extend it through RSS feeds, you can have a mobile version of your website, you can have a print version of your website. It's beneficial to you and it's beneficial to your audience to actually make sure your content is format neutral. One of the

great byproducts of this, when you're migrating your site and you're redesigning it, you can take that format neutral content and easily migrate without having to deal with font tags or other styles embedded directly in your content.

[Slide Bullet on Screen: No formatting in your HTML mark-up]

And with that, make sure that you have no formatting in your HTML markup. I don't want to see any font tags. I don't want to see bold tags. I don't want to see italic tags. Keep all that stuff out of your content. Let all your formatting be handled with CSS. If you have to stylize specific text elements, use classes or widely supported WCAG and W3C endorsed tags.

[Slide Bullet on Screen: Text: size, color, contrast, max width, no full justification]

The last piece of this is with regard to text size and color and contrast. Make sure you're following the recommendations of WCAG and you've got a readable size, there's enough contrast between the background and foreground, that you've set a maximum width within your webpage. Don't let your web page scroll out to kingdom come. Set a maximum width. For example, there is a subset of users where all they can see is a very narrow piece of the content and they have to scroll, scroll, scroll, scroll, scroll to read through the content. If they have to just keep scrolling to kingdom come, it's very difficult and cumbersome to navigate through content. So provide a maximum width and also don't use the full justification within your font. That also makes it difficult to read because of the extra spaces generated between words. So use your standard left justification for your body content.

10. Site and process context

And the last issue, number 10: site and process context.

[Slide Bullet on Screen: Breadcrumb navigation]

First of all, make sure you're providing breadcrumb navigation if you have a larger site. We touched on that a little bit earlier.

[Slide Bullet on Screen: Use title and heading tags to convey site context]

Use your title and heading tags to actually convey site context.

[Slide Bullet on Screen: Provide a sitemap]

Also, if you have a larger site, consider using a site map. Site maps will give the user an at-a-glance view of the context of your site and how it's structured and architected.

[Slide Bullet on Screen: Use a search function]

Also consider using a search. You want to make sure the user is able to quickly drill down to content that is pertinent to them.

[Slide Bullet on Screen: Forms: if multiple steps, indicate context]

And lastly, if you've got forms or a Web application that have steps involved or there's a sequence or an order, make sure the you're conveying the context or the order of where they are in those steps. A good example is say TurboTax and how they give you an at-a-glance, step-by-step view where you are in the process. Make sure that your web forms or your web applications that have those sorts of processes or steps that they know exactly where they are in the process.

Resources, Tools and Conclusion

Alright, in this last section we're going to learn about some resources that you can use for testing and help you gain a better understanding of accessibility.

[Slide Bullet on Screen: WCAG 2.0: http://www.w3.org/TR/WCAG20/]

First of all, of course, there's WCAG 2.0 and that's been released by the W3C. That's just fundamental. You have to know this. Take some time, maybe a few minutes out of your day, for a period of a week or two, and just with this first link (noted above), visit WCAG 2.0 and read through some of the principles.

[Slide Bullet on Screen: WebAIM accessibility testing: http://wave.webaim.org/]

Sometimes that can be a little bit more intimidating. There are great links and resources at Webaim.org. It can help you break down the voluminous document on WCAG 2.0.

[Slide Bullet on Screen: How people with disabilities use the web: http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/]

There's an article that's also been released by the W3C (noted above) that I highly recommend called "How people with disabilities use the web." This gives you a little bit of an empathy and understanding for the situations that other people have in terms of how they interact with the web and how they surf. As a web designer myself, you reach a point where you design and you build with your own understanding in your own worldview but until you actually open up your view and understand what it's like to basically live and function within someone else's shoes or someone else's capabilities or someone else's environment. Until you understand that, it's really hard to really get a grasp as to why WCAG is important. So

spend some time in that article and read through it. It gives some great case studies in terms of how different people use the Web.

[Slide Bullet on Screen: Examples used in this presentation: http://accessibility.designbymichael.com/examples/]

[Slide Bullet on Screen: Links to presentation files: http://accessibility.designbymichael.com/presentations/]

Also, I've got some examples used in this presentation (noted above). You can find them available on the web, as well as some tools and testing.

[Slide Bullet on Screen: Firefox: https://addons.mozilla.org/en-US/firefox/extensions/web-development/]

Here are some resources from Firefox (noted above). There's a couple of different plug-ins that I highly recommend. The Web developer toolbar is essential for debugging your HTML or your CSS or other accessibility issues.

There's a WCAG contrast checker that actually gives you testing values for testing text color or other elements against their background, as well as the FANGS screen reader emulator which is great for getting a text-only view of what your website is like.

[Slide Bullet on Screen: Safari: https://extensions.apple.com/#developer]

If Safari is your flavor, there's a few plug-ins and add-ons which give you better HTML view of your source code.

Firebug Lite, Unicorn HTML, a single CSS validator for validating the code as well as the plug-in Validator which validates against WCAG principles.

[Slide Bullet on Screen: Google Chrome: https://chrome.google.com/extensions/featured/web_dev?hl=en-US]

If Google Chrome is your flavor, there are a few tools like Web developer, View selection source and HTML validator.

If you're an Internet Explorer-only shop, natively in IE 7 and up, there is the developer toolbar and there are some things that actually will help you test. If you use Visual Studio 2010, there are accessibility testing tools with them as well.

[Slide Bullet on Screen: Color Contract Analyser]

Client-side, there are some great tools and applications that I highly recommend. First, Color Contrast Analyser (noted above). There's a version for both Windows and Mac and it's great. You can use an eyedropper where you can test the color of your text and then you can use the eyedropper to test the background color and actually gives you the output of where it conforms with AA and AAA and how it conforms with like the large text view or the small standard copies. It's a great app.

[Slide Bullet on Screen: Total Validator:

- Basic desktop application (free, single-page testing)
- Pro desktop application (inexpensive, site-wide testing)
- Web-based testing (free single-page testing)
- Firefox add-on (free, single-page testing)]

TotalValidator is also another product that I've used (noted above). There's a free version as well as the one that you can pay for with added features on the Pro version. But they have a basic version which is the free one, the Pro and there's a web-based tool of it that you can test, as well as a Firefox add-on which is great. You can do a single test page to make sure it's not only accessible, but you can test the validity of your HTML. You can even test spelling and check for broken links. Great, all in one app.

[Slide Bullet on Screen: WebbIE]

WebbIE is also a great app if you've got a PC available to you (noted above). There's a free text-only browser and it's a great way for getting the perspective of how a screen reader might actually go through and read and render out your HTML or JavaScript or what not. So I use it all the time if I've got a gray area with like jQuery or JavaScript and I want to know how it functions. It's a great, great app.

[Slide Bullet on Screen: System Access To Go]

System Access To Go is a Windows-based screen reader utility. If you don't have the funds and access to Jaws, and Jaws is kind of pricey, System Access To Go is good. There's also another one called Thunder. Thunder is also a PC-based one. The Mac has natively within it accessibility tools and is called Voiceover and works natively with Safari. It's also another great tool to get perspective of what it's like to browse the web with a screen reader. In fact, an eye-opening experience would just be to put a blindfold on for 5 - 10 minutes and then try to surf the web with a screen reader. It's really just an enlightening experience and helps further give you more empathy about what it's like to browse the web with a screen reader only.

Questions

[Slide Content on Screen:

Michael Tangen | web interface designer-developer

Office of Enterprise Technology]

So that's all in terms of the top 10 presentation. I'm very thrilled and grateful that you all came and to all of you in the audience, I'm grateful that you took the time to watch this presentation. If you've got questions, you can reach out to me. Here's my e-mail address as well as my direct line if you need to reach me at all: michael.tangen@state.mn.us; (651) 201-1045.

Contact us

If you have questions about this change, please contact the OET Service Desk:
651-297-1111
Service.Desk@state.mn.us
www.oet.state.mn.us