

Meet A Member of the Team:



Christopher Walker

Christopher Walker's role in the VA Section 508 Office is to work directly with project teams and help them meet the Section 508 requirements.

Born in Missouri, Chris enlisted in the Marine Corps at age seventeen and served for eight years, earning the rank of Sergeant. His assignments included four years as a computer programmer and four years as an interrogator.

After leaving military service in order to devote more time to raising his two children, Chris attended college and earned a degree in computer science.

He now lives in South Texas, where he says he loves the fact that there isn't any snow.

He has worked on a wide range of software projects including deep water undersea pipe fitting and control monitoring, a NASA satellite that went to Jupiter, software for the Center For Nuclear Waste Research Analysis and water monitoring quality software for the city of San Antonio.

How did you get interested in Section 508?

About five years ago I started as a contractor for VA, working on pharmacy-related applications. That's when I first learned about Section 508. Until then much of the work I did for the government fell under the Section 508 exception that applies to National Security.

I didn't really think about anyone using pharmacy applications having a disability. They'd need to be sighted in order to work with the pills.

But then I found out there were people with disabilities working in VA who needed to use certain aspects of the

INSIDE THIS ISSUE

Meet A Member of the Team	1
Quick Tip: Save It As A .Doc	2
Section 508 Requirements and Your Videos	3
Creating Accessible Infographics	4
Plain Language Provisions: User Interface Elements	5
Twitter Releases Accessible Images	7
CSUN 2016 Recap	
Federal Accessibility in Focus	8
Get Onboard!	8

application. It really opened my eyes. There were people out there using applications that I wouldn't think had to be 508 compliant.

I started to put time and effort into understanding what it means to be accessible in terms of Section 508. Along with thinking about people who are blind, I started thinking about people who are deaf. I had written some quality software for oil pipelines that was very dependent on tones. Tones and color were everything in that software. We had beeps and alerts. The tone of an alert would tell how critical the situation was, and how quick the response needed to be. A blind or deaf user could certainly do the job of monitoring that pipeline, but they wouldn't be able to use the software as it was written.

This was for a commercial company, so 508 didn't apply, but when I look back, it wouldn't have been that difficult to use other means to get the same information across.

It's one of those things where, if you think about it early on, your design can pretty easily encompass being

accessible to everyone. But if you don't think about it early on, once you start down the path, it's really hard to get that path changed.

When an opportunity to work in the Section 508 Office came up I decided to put my hat in the ring. I got the job and I'm really enjoying it.

How can the Section 508 Office help project teams?

By making the requirements easy to understand. There is no "easy button" but we can give easier answers. Section 508 isn't always easy, but it doesn't always have to be hard either. We have real successes when we get directly involved with project teams. We can show project teams what to look for and how to test their applications themselves before their project gets audited by the 508 Office. When we have an opportunity to meet and work with project teams early on, compliance goes up. We've had projects that started out with a large number of defects. Then we worked with the teams, showed them where the barriers were, and now the applications are much more accessible.

What advice would you like to give to developers?

When VA is first onboarding contractors or employees to design or build something, it would be really nice to say, "508 isn't just a line item on your checklist. It's something you need to put into your early design efforts."

There are some things developers can do to start checking 508 compliance issues, like using the keyboard to go through their application and make sure they can access all the content. They can load a screen reader and make sure everything gets read. We can show developers other ways to test their projects that will help them catch some of the major 508 compliance issues.

VA serves a large community of Veterans with disabilities. Also, employees working directly with Veterans need to be able to access applications to do their jobs.

Are there any last thoughts you'd like to share?

I believe being committed to something is important. Is what I'm doing going to improve somebody's life? If not, why am I doing it? When we make an application out on a VA site 508 compliant and easy to use, those are things that directly benefit Veterans. I think that's compelling.

Quick Tip: Save As a .Doc



You may have noticed that if you convert your Word document to PDF using the recommended approach of exporting through the Acrobat tab, and you examine the tag structure in the Tags pane, all of your images and graphics in the document have been clustered to the top at the beginning of the document. Having the application yank them from their intended location in the document completely messes up the intended reading order, a critical aspect of accessibility for documents. You then need to move those Figure tags carefully back to their places. In a large document with lots of images, this can be time consuming and frustrating, but here's a simple workaround.

After you've done all you can to make your Word document compliant - run the accessibility checker, and referred to techniques found in the VA Section 508 [Creating Accessible Word Docs with Microsoft 2010](#) tutorials - simply save your Word .docx file as an older format .doc file. Ensure that you have fully remediated the .docx file before saving as .doc because you will not be able to use the accessibility checker on the .doc format.

Then proceed with exporting the .doc file to PDF. When you examine the Tags pane of the resulting file, you'll see that the Figure tags for your images and graphics are all in their intended location; saving you from having to move them painstakingly back to their proper homes in your document.

So with image-heavy Word documents, just insert that one step of saving your compliant .docx file as a compliant .doc file *before* you export it to PDF into your workflow, and your images should remain where you put them in the first place.

Section 508 Requirements and Your Videos

Did you know that there are Section 508 requirements for videos? Here are some tips and best practices to help you meet those requirements and make videos more accessible to your entire audience.

CAPTIONS

All training and information videos that support the agency's mission must be captioned if they include speech or other audio needed in order to understand the content.

Captions are text versions of the spoken word and any other meaningful audio content contained in a video. Although they are most commonly used by people who are Deaf or hard of hearing, they may also make video content more accessible to people who prefer to read the spoken words rather than listening to them.

There are two types of captions. Closed captions (soft captions) can be turned on or off by the user, while open captions (hard captions) are a permanent part of the picture.

Open caption text can be added to the video directly. The subtitles can be overlaid onto the video using video editing software. Users do not have the ability to control the size and appearance of the captions.

Closed captions are created by using software to create the text and synchronize it with the video, and then exporting the captions in the desired format. Most captioning software can export the captions into several

different types of files. Generally, the user is able to turn captions on and off and change the size and appearance of the text.

Captions must be synchronized so that the text content appears at approximately the same time as the audio content. If you have ever watched a film where the subtitles don't match up with the action on the screen, you probably have a sense of how difficult it can be to keep track of what's being said, and why synchronized captions are a Section 508 requirement.

Several vendors provide high-volume captioning services. There are also several web-based and desktop captioning tools that allow developers to create their own captions for streaming videos. Visit the VA Section 508 website for a list of [Caption Resources](#).

TRANSCRIPTS

For audio-only content such as podcasts, a transcript is sufficient to meet the Section 508 requirements. Transcripts may also be beneficial to members of your audience who, for a variety of reasons, may prefer to read your presentation rather than watch the video. Transcripts also allow potential viewers to locate your video using search engines.

However, it is important to remember that Section 508 requires captions for informational and training videos which include both audio and visual content and support the agency's mission.

AUDIO DESCRIPTION

Section 508 specifically requires audio description for training and informational videos that support the agency's mission if they also contain visual information that is necessary in order to understand the content of the video.

Audio description consists of explanations of meaningful visual content, spoken either as part of the narration or inserted during pauses in the original narration. The easiest way to create audio description is, when possible, for the presenter or narrator to describe actions and information as part of the original presentation. For example, having the presenter say, "I'll click on the green activate button" is easier than it would be to add that information as audio description after the video is finished. This approach works well for videos that have not yet been created, especially if adding description to the narration doesn't interrupt the presentation flow.

A second approach is to provide two copies of the video – one with a standard audio track and another with audio description.

A third approach is to provide an additional audio track that can be toggled on and off (much like captions) and plays at the same time as the video.

In cases where none of these approaches is feasible, a transcript can be used to meet this requirement. The transcript should include explanations of meaningful visual content as well as

the full text of the audio presentation and any other meaningful audio cues.

For more information, visit [Audio Description For Multimedia](#)* on our website.

NO DISRUPTION OF ACCESSIBILITY FEATURES

Section 508 requires applications not to disrupt or disable activated accessibility features. Because videos that play automatically can make it difficult or impossible to hear a screen reader, videos should start playing only when they are activated by the user.

ACCESSIBLE PLAYER CONTROLS

Once your video meets the Section 508 requirements, you need to be sure that the media player does too. All player controls including play/pause, volume, forward/rewind, and captions on/off must receive focus and be operable using a keyboard, not just a mouse.

Information about each control must also be available to assistive technology. For example, when a screen reader user tabs to a control, the screen reader should identify what the control is, what it does and provide

information about its current status – on or off, activated or not, expanded or collapsed and so on.

You can learn more about accessible controls in the “Section 508 Provisions in Plain Language” article in this issue.

Planning ahead while you’re creating or choosing videos can make it much easier to meet the Section 508 requirements. If you have questions after reviewing the resources listed in this article, please contact Section508@va.gov

Creating Accessible Infographics

Infographics are popular for several reasons – they can quickly convey complex bits of information, and visually illustrate key relationships between data sets and display large numbers and concepts in a simple way. The sighted user can quickly scan the information and move on. So how do we make those images accessible? Two ways.

1. Provide a text alternative on the page that conveys the same meaning and info as the image file (easy)
2. Use HTML/CSS to create your infographic rather than use an image file (harder)

WHAT TO DO:

The infographic is essentially an image – so choose a color palette that meets color contrast requirements and don’t rely on color alone to convey the information. For example, “There are 21.8 Million Veterans (20.2 Million Male/ 1.6 Million Female) in the United States” is a better choice than “There are 21.8 Million Veterans in the United States, Male Veterans shown in blue, Female Veterans in red”.

Create a long text alternative that presents the same info and group the text into a logical hierarchy. Provide the appropriate heading markup to support that structure. So in my example, if your Veteran infographic has depictions

of different data such as number of Veterans, ethnic or gender breakdown, length of service, employment data etc, then make those categories into a logical heading structure.

And last, but probably most important, decide whether you are going to include the text alternative on the same page as the infographic or on a separate page. Provide a link before the infographic to the text alternative with meaningful link text such as “text alternative for Veteran data infographic”. Although you can use the longdesc attribute to direct users to the location of text alternative, this method is not recommended because the longdesc attribute is only visible to screen reader users, meaning that other users who may require the long text alternative will be unable to access that information.

RESOURCES

- [How to Create Accessible Infographics](#)
- [Web Accessibility for Designers](#)
- [Web Accessibility for Designers \(HTML/CSS accessible example\)](#)

**Links designated with an asterisk are available to VA internal users only.*

Plain Language Provisions: User Interface Elements

This is the first in a series of articles called *Plain Language Provisions*, where we will explain some of the commonly misunderstood Section 508 requirements. The requirements are part of the Section 508 standards, which are available on the United States Access Board website. The requirements are listed in lettered paragraphs under Subpart B – Technical Standards, Subpart C – Functional Performance Criteria, and Subpart D – Information, Documentation and Support. They are called provisions because, legally speaking, they represent an agreement between a federal agency and the people or organizations responsible for the Electronic and Information Technology (EIT) that it develops, procures, maintains, or uses.

Provision §1194.21(d) is one of the most important in making EIT accessible, and it states:

Sufficient information about a user interface element including the identity, operation and state of the element shall be available to assistive technology. When an image represents a program element, the information conveyed by the image must also be available in text.

So what does that mean?

For every interactive piece of an application, (including buttons, links, checkboxes, menu-items, dropdown boxes, and edit boxes) whether that application is a desktop application, a Web application, or a mobile application, certain information needs to be available to assistive technology (AT).

HOW DO YOU MAKE INFORMATION AVAILABLE TO ASSISTIVE TECHNOLOGY?

In order to understand what this requirement is stating, it helps to know a little bit of computer history. Screen readers are primarily used by people with visual impairments. They speak out loud information presented onscreen, using synthetic speech. When information was presented as simple text in a Command-Line Interface (CLI), such as the MS-DOS operating system, providing information to a screen reader was relatively easy.

```
Displays a list of files and subdirect
DIR [drive:][path][filename] [/P] [/W]
  [/S] [/B] [/L] [/C[H]]

[drive:][path][filename] Specifies
/P      Pauses after each screenful
/W      Uses wide list format.
/A      Displays files with specifie
attribs D Directories R Read-o
        S System files A Files
/O      List by files in sorted orde
sortord N By name (alphabetic)
        E By extension (alphabeti
        G Group directories first
```

Figure 1: MS-DOS screen

All a screen reader had to do was intercept the same text that was being displayed on the monitor, and read it back in the same linear fashion. However, when graphical user interfaces (GUI) such as those popularized by Microsoft Windows 95 went mainstream, suddenly screens were filled with menus and icons and buttons. Those user interface elements were laid out spatially – not necessarily from left-to-right, top-to-bottom. More importantly, not only did the way computers look change, but so did the way we could interact with them. In MS-DOS, interaction was purely textual. Users typed commands in, and they confirmed them with the Enter key. If they were lucky, they could delete or insert characters that were mistyped. In Windows, users can press buttons and check checkboxes and open applications and highlight files.

Each of these interactions requires different keyboard input from the user: for instance, SPACE BAR highlights a file, ENTER opens an application and ALT+DOWN ARROW opens a dropdown menu. Today, most users don't know those commands by heart because they take that interaction for granted. Along with the GUI, Windows 95 also popularized the mouse. The mouse unified interaction: with anything on your screen, no matter what it was or how it worked, all you had to do was click it and something would happen.

But what if you couldn't use a mouse? How could AT tell you what something was, and what to do with it?

ENTER THE ACCESSIBILITY API

Early AT used complicated heuristics to attempt the same tricks it used to display a CLI: trying to guess where things on the screen were drawn and how big they were. Based on that, it could attempt to tell the user this looks like a button or this is probably a dialog box. But as GUIs became increasingly complex, it became critical for designers and developers to have a reliable way to tell assistive technology intentionally what something was.

The solution was an Application Programming Interface (API) that AT can query in order to know all of the important things about GUI elements. Today, most major platforms used inside the VA (and in the public by Veterans and their families) include accessibility APIs.

The accessibility API parses code written by developers and exposes several pieces of information. For example, consider the following code for an HTML element:

```
<input type= "checkbox" id="org">
<label for= "org">VBA</label>
```

- The type of element is Checkbox. That type is specified by the <input> element.
- Its Name is VBA. The name comes from the <label> element.

A screen reader (or anything that uses the accessibility API, such as a Web browser, speech recognition software, or a screen magnifier) will know that the piece of the screen represented by that code is a Checkbox called VBA. Upon encountering this UI element, an AT user will know two very important things about it:

- What it is (a checkbox)
- What it means ("VBA")

An element's structure refers to how it was built. Structure is critical to knowing how to interact with something, because an element's structure implies its interaction. In the code above, a user who knows the element is a checkbox knows that in order to check it, they hit SPACE BAR. If the VBA code above were for a link, a user would not

check it with the space bar, they would activate or follow it with the ENTER key.

Different (key)strokes for different folks – or, UI elements, whatever the case may be.

WHEN IS A CHECKBOX NOT REALLY A CHECKBOX?

Using <input type= checkbox> guarantees that developers will create something that is, in fact, a checkbox. Using code that is less structurally rich, a developer could create something that says VBA on it. For instance:

```
<span id= "checkbox">VBA
<span id= "icon"></span></span>
```

Using CSS, the developer could make that thing look like a checkbox and could make it clickable by using a JavaScript OnClick event. But even though it looks like a checkbox, and clicks like a checkbox, it is not actually a checkbox. The element is semantically neutral. It has no role. Without a <label>, it has no name.

Upon encountering this element, someone using AT would have no idea what it is or what to do with it. Even if the AT could magically guess what it was, the is not operable with the keyboard, nor does it tell the user whether the VBA box is checked or unchecked. Using <input type= "checkbox"> gives all of that information effortlessly to the Accessibility API.

TESTING STRATEGIES

1194.21(d) requires that the identity, operation and state of UI elements be available to the assistive technology. Let's assume we wanted to check that information for part of Microsoft Office 2010.

"Cut" is grayed out in the photo, but what is it really? Is it a menu item, a button, or something completely different? And does the name "Cut" match the visible label? Also, it's grayed out.

Visually, users know that means it's unusable – but how is that information conveyed to AT?

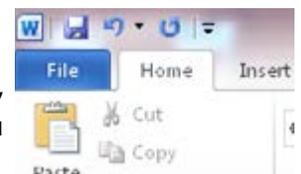


Figure 2: The copy button is grayed out.

One way to check that is with Object Inspector, which is part of the Windows SDK. Inspecting “Cut” reveals the following information:

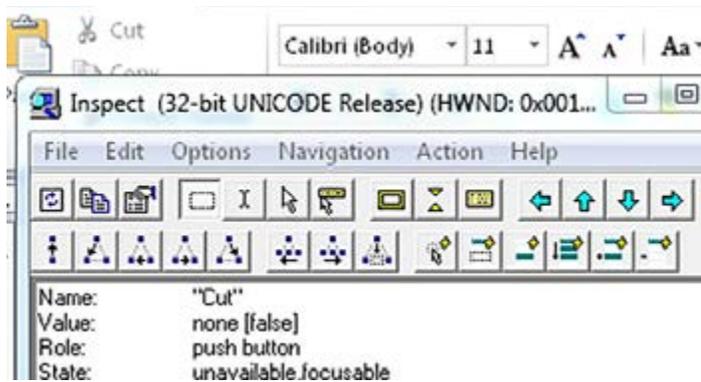


Figure 3: Object Inspector Overlay

Using Object Inspector, it is plain that the Cut icon is a button called Cut. It is programmatically unavailable. This information will be communicated consistently to assistive technology. Upon encountering this control, a screen reader might speak “cut button unavailable” or “cut button dimmed”. In any case, it is obvious that the UI exposes sufficient information: it is, in fact a button. A user could press it and text would be Cut. But the button cannot be pressed at the moment, because no text is selected.

Ultimately, this provision requires that all of the visual cues available during an interaction be available programmatically as well.

Using Object Inspector is quite technical. If you prefer functional test, you could use assistive technology and navigate to (via the tab key, or via a tap if using a mobile

screen reader) and hear what the screen reader speaks. Upon hearing the information, do you know what can be done with this element?

WORTH A THOUSAND WORDS

Last but not least, 1194.21(d) requires that the information conveyed by images be available in text. Consider the magnifying glass that is part of a web browser’s address bar, like this:



Figure 4: That magnifying glass isn’t for Sherlock Holmes

Intuitively, many people who can see know that the magnifying glass icon means Search. Indeed, you can perform a Google search by typing a search query and pressing enter. How would AT know what that magnifying glass does? A quick listen with a screen reader or a quick peek with object inspector reveals that the identity (or name) of that control is Search. Moreover, it reveals that the Search magnifying glass is a menu item – something people who can see would understand by seeing the nearby dropdown arrow.

In this case, it is obvious that enough information is available to the accessibility API to know what the magnifying glass is (a menu) and what it does (it searches). The magnifying glass is 508 conformant. How does your application compare?



Twitter Releases Accessible Images

About a month ago, Twitter made an important step by releasing an announcement focused on accessible images for everyone. Mobile Twitter fans using their iOS and Android apps now have the ability to add alternative text or descriptions to photos posted on the social media platform. Select “compose image description” in the app’s accessibility settings to enable this feature. Then when you add an image to a tweet, an “add description” button will display on the thumbnail in the composer. Tap the button to add a description up to 420 characters. The alternative text will be available to assistive technologies such as screen readers and Braille displays. The ability to add descriptions to photos is currently not available through desktop clients.

Resource: [Accessible Images for Everyone, https://blog.twitter.com/2016/accessible-images-for-everyone](https://blog.twitter.com/2016/accessible-images-for-everyone)

CSUN 2016 Recap: Federal Accessibility in Focus

A member of the VA Section 508 Office Team was able to attend the 31st Annual International Technologies and Disabilities Conference in San Diego, CA from March 23 – March 25, 2016. The conference (called CSUN, a reflection of its beginnings at the California State University Northridge) is the largest gathering of accessibility professionals from the public, private, educational, healthcare, and non-profit sectors. At the end of each conference, slides from the sessions are crowd-sourced by attendees and compiled as part of *The Great Big List*. Feel free to peruse that list. It represents some of the most current and cutting-edge accessibility training available today.

The first stop was a session called, “Why WCAG?,” helping to explain some of the common mistakes that people and organizations make when trying to implement the Web Content Accessibility Guidelines (WCAG 2.0). WCAG 2.0 is an international accessibility standard that has been used by the U.S. Department of Justice (DOJ) as the benchmark for nearly every accessibility lawsuit settled in the United States since 2006. As we have covered in previous issues, WCAG will be incorporated by reference into Section 508 as part of the impending Section 508 Refresh.

Accessibility at the state and federal government levels was a popular topic at CSUN. Our colleagues across Government led sessions, including:

- The Central Intelligence Agency shared its approach to monitoring 508 compliance, and creating test scripts using Cucumber.io
- The National Association of Chief Information Officers shared updates on incorporating accessibility requirements into state government procurement processes.
- The Food and Drug Administration shared its approach for creating 508 compliant SharePoint 2010 sites.
- Several organizations shared their thoughts on recent rulings and settlements from DOJ that applied the Americans with Disabilities Act to web and software applications, and used WCAG 2.0 A and AA as compliance benchmarks.
- The Department of Labor’s Office of Disability Employment Policy (ODEP) shared its approach to working with vendors to achieve accessible product design.
- ODEP and DOT spoke on how innovations in accessible transportation will increase access to employment opportunities for people with disabilities. Elsewhere, Google gave an update on its self-driving car project and spoke about its engagement with DOT, state lawmakers, and some of the regulatory challenges involved in rolling out autonomous vehicles.
- The State of Minnesota explained how it meets its requirements to make Government information shared on social media accessible.

If you have questions about meeting Section 508 requirements at VA, please feel free to reach out to us at Section508@va.gov.

Get Onboard!

It is now possible to be alerted when a new edition of the 508 XPress becomes available. Just visit www.section508.va.gov/support/newsletter and activate the link to subscribe to our list.

Visit the VA Section 508 website to review Section 508 checklists; training materials for developing accessible content in Flash, HTML, Word, PDF and PowerPoint; register for courses and to locate additional resources.

Internet: www.section508.va.gov || Intranet: vawww.section508.va.gov*

**Links designated with an asterisk are available to VA internal users only.*