

It's Time to Update 508

The United States Access Board published the [Section 508 standards](#) in the Federal Register in December 2000. Since then, the Section 508 standards have specified the technical and functional characteristics that Electronic and Information Technology (EIT) needs to have whenever it is developed, procured, used or maintained by federal agencies.

The last 15 years have seen an incredible uptake of technology in the United States and abroad. Websites have evolved from static HTML documents to dynamic, scripted, robust applications as functional as desktop software applications of old. Mobile computing is commonplace. The Department of Veterans Affairs engages with Veterans and the public as deftly on [Facebook](#), [Twitter](#), [Youtube](#) and [Flickr](#) as it does through traditional walk-in appointments or call centers. Recognizing the need for a revised set of standards to reflect the modern use of technology both in Government enterprises and in the daily lives of citizens, the Access Board released the first Advanced Notice of Proposed Rulemaking for revised Section 508 standards on March 22, 2010.

The 2010 release was the first time the public officially heard about what is known as the “508 Refresh.” The [508 Refresh](#) will re-define the technical and functional characteristics of 508 conformant technology, without changing the Section 508 law itself. The Access Board is given the authority to do this under federal law, [Section 508 of the Rehabilitation Act](#) (29 U.S.C. 794d), as amended by the Workforce Investment Act of 1998

INSIDE THIS ISSUE

It's Time to Update 508	1
Quick Tip: What's In A List?	2
Mobile 508, Native, and Hybrid: A Section 508 Perspective	3
ARIA's Hot Rolls (Roles)	5
Is That Content Really Hidden?	5
Get Onboard!	6

(P.L. 105-220). Although the Access Board leads the Refresh effort, it is done in collaboration with the U.S. Office of Management and Budget and members of the public who may [submit comments formally through Regulations.gov](#).

SECTION 508

After nearly five years of review, the Access Board published a Notice of Proposed Rulemaking (NPRM) alongside the [full text of the 508 Refresh](#) on February 18, 2015. Among the highlights are:

- Removal of “product categories,” so that applicable standards are determined based on product features and functionality, instead of what type of platform it was built on. For example, whether you are creating a website, a desktop application, a mobile application, or an electronic document, it needs to be navigable using a keyboard.
- Harmonization of the Section 508 standards with international accessibility standards, including the [Web Content Accessibility Guidelines](#)

(WCAG) 2.0 and [PDF/UA-1](#). This also includes using the term Information and Communication Technology (ICT) to replace EIT to mean “technology that must meet the Section 508 standards.”

- Clarification on which types of electronic content need to meet the Section 508 standards include: all public-facing content, all forms, all templates; and all training materials. For example, an emergency notification, an initial or final decision adjudicating an administrative claim or proceeding, an internal or external program or policy announcement, a notice of benefits, program eligibility, employment opportunity, or personnel action; a formal acknowledgement or receipt; a questionnaire or survey; a template or form; and educational materials would all be covered.
- A requirement for “[Real-Time Text](#) (RTT),” meaning that any product that allows for real-time, two-way voice communication (such as telephones and Voice Over Internet Protocol systems like Microsoft Lync, Skype, Google Hangouts, or Apple Facetime) must also allow for the written words to be transmitted on a character-by-character basis as they are typed, rather than as a single block of text once transmission is complete.

It is unknown when the final rule will be published, but the Access Board is currently considering adding six months to the effective date so that the Federal Acquisition Regulations (FAR) Council may have sufficient time to update the FAR in order to help agencies achieve Section 508 compliance. Even so, we do not anticipate another years-long delay before the 508 Refresh is official.

VA Section 508 is following the progress of the Refresh closely. We will continue to provide guidance, training and support as the Refresh moves forward.

www.section508.va.gov

Quick Tip: What’s In A List?

This quick tip applies to HTML, PDFs, and Word documents. Certain page structures (like lists) screen readers are capable of recognizing and communicating to the user. A screen reader can identify lists and list items (for both ordered and unordered lists) if they are tagged and coded correctly. This is especially important for nested lists and those used for outline structures, because that way a user can tell both which item within the list or sublist they are on, and the level or sublevel. A screen reader can tell the user that there is a list of 7 items, and then correctly identify items within that list.

So it is best to use the tools and markup that are available for creating lists. This means using the correct markup in HTML, lists styles in Word, and list tags in PDF.

HTML: HTML list elements should only be used to create list structures, and should not be used for formatting effects such as indentation. Screen reader users may mistakenly interpret these items as list structures and may be disoriented when ``, ``, or `<dl>` elements are defined without ``, `<dt>` or `<dd>` elements.

PDF: Pay attention to lists that span across multiple pages. Even though a list continues on to another page, it is important that in the tag structure those list items are part of the original list and not a list of their own. Attention to detail of these items enables users to associate related content and know the total number of related items.

WORD: When creating lists and sublists, try to avoid using graphics or non-standard characters as bullets. Screen readers are sometimes confused by these characters and will not automatically understand them as bullet characters. Using the standard bullet characters whenever possible along with assigning styles to your lists give you accessible multi-level lists.

Review more guidance for these topics online at www.section508.va.gov/support/.

Mobile Web, Native, and Hybrid: A Section 508 Perspective

Over the past couple of years our Office has had the privilege of testing a variety of Mobile Content for Section 508 Conformance. As our testing methods and processes are maturing, we're beginning to gain a perspective on areas where our customers might benefit from an explanation of the types of content we test, how these types of content differ from each other when accessed by users of mobile technologies, and the kinds of challenges and opportunities that arise when varying types of Apps are tested within tablet and mobile handset environments.

For instance, we may have certified a specific App that is now being enhanced and laid out in a very different manner than the original content, so we must retest the content to ensure that all applicable Section 508 requirements are being adhered to during the development process. This article will explain the difference between Mobile Web, Native, and Hybrid Apps, discuss their differences as well as their benefits and drawbacks, and provide project managers for or developers of Mobile Apps some strategies for moving forward with current and future mobile development efforts for the VA.

WHAT ARE THE DIFFERENCES?

To begin with, let's define these three types of mobile content. "Mobile Web" is best explained as web content developed in HTML5 designed to be displayed and accessed on a tablet or mobile handset. The device's Browser (i.e. Safari, Chrome or Firefox) uses the web contents underlying code to render the text so that the information is resized to fit the screen on a given mobile device. The various web elements such as headings, links, and form controls are easily navigable by the Browser's built-in functionality and the assistive technology built into these devices (Voiceover, iOS, and Talkback, Android) is able to leverage off their alternative screen gestures to allow individuals to interact with this information.

A Native App is designed to be accessed through the device's "App Store". It's written in the device's native

code, thus giving the App a different look and feel than the web content displayed via Mobile Web. In short, native Apps take on the characteristics of their respective operating systems, and allow developers the flexibility of tying into the various components of a mobile device such as "location services", GPS know-how, the camera, and other distinct characteristics within the operating system that define the behaviors of a given iOS or Android Device.

Hybrid Apps are a combination of Mobile Web and Native App. They are developed in HTML 5, CSS and JavaScript, with the developer using a utility such as Cordova to run the App inside a native container and use the device's browser engine, not the browser. These containers allow the HTML and JavaScript to hook into the camera and GPS technology of specific devices. The App can be installed on one or both iOS or Android platforms, depending on the scope of the development effort. Hybrid Apps look and behave a little differently than native Apps in that the Browser is embedded in the iOS or Android App. This means that certain controls such as a "back" button may behave similarly to the "back" button in a Browser like Safari rather than the operating system's built-in control.

WHAT ARE THE BENEFITS?

The greatest benefit of developing Mobile web content is that it's probably the easiest of the development efforts. If there is current web content that exists, and the idea is to make this content optimal for mobile Browsing, it doesn't take a great deal of development effort to make this happen. It's the perfect solution when the intent is to provide a means for its users to obtain, search for, and interact with helpful information.

The primary benefit for developing an App native to the device's operating system is that not only can you take full advantage of all that the device has to offer, but you can create an App that really looks and feels like the device's environment. You can also leverage off the APIs and best practices that Apple and Google offer to ensure that your App is accessible and

Section 508 Compliant. Hybrid Apps offer developers the Mobile web approach for development with the ability to wrap the App for a desired operating system, thus giving the project team the advantage of a more intuitive, less costly development effort combined with the robustness of a Mobile App that can be downloaded from the App Store. Developing Apps for multiple platforms as well as maintaining and enhancing them are most efficiently achieved by embracing the approach when creating a hybrid App. It's also a less expensive means of getting your App listed in the device's App Store due to the reduced time for development.



The major drawback to the development of a Hybrid App is that it may be difficult to convert content designed in HTML to fully reflect all of the best practices that either Apple or Google offer when wrapping the content into a given App. VA Section 508 has run into a few accessibility barriers, when a given control isn't behaving quite as it should once the web content has been rendered as an App. So at times, a fair amount of research and creativity may need to occur to bring this content into compliance that might not be as much of an issue for developers of Native Apps.

WHAT ARE THE DRAWBACKS?

The most significant drawback to Mobile Web content is that it is designed to be accessed through a Browser. This is fine if the intent is for the content to provide information as well as the ability to search for information. Such content must be accessed while the user is connected to the Web, although many mobile webpages have a “reader” utility which allows you to save and reformat content for offline viewing. Again, if the intent is to provide information, Mobile Web is definitely the way to go—at the risk of restating the obvious; Mobile Web content is not accessible through a device's App Store. The device's Browser must be engaged and the appropriate web address must be entered; users do have the option to bookmark this content to find it more efficiently for future visits.

The biggest drawback for developing a Native App can be cost—it costs a fair amount of money to not only develop an App, but maintain it and appropriately plan for future development efforts. Also, if the intent of a Native App is to provide information, it's a little more difficult for this information to be found when searching the App Store rather than the World Wide Web. Lastly, to take a Native iOS App and create a similar Native App for Google requires that a new process and considerations be implemented from the ground up.

MOVING FORWARD WITH SECTION 508 CONSIDERATIONS

In the past few months VA Section 508 has received a few inquiries regarding the need to have previously certified Mobile Web content retested for Section 508 compliance once this content has been converted to a Hybrid App. In short, the answer is yes. As this article has pointed out, there are a number of differences between how Mobile Web and actual Mobile App content behave within their respective environments. Mobile Web content consists of Browser-specific controls and web content includes web controls. An App includes controls for the functionality of the App, App content, and tabs that allow for different screens of the App to be accessed. A common problem we encounter occurs when converting from HTML5 to a hybrid App. Many of the App controls are rendered as hyperlinks within the App. While this is acceptable when navigating to different webpages, even within the specific web content being tested, a link should only be used within a Hybrid App to navigate the user outside the confines of the App; alternatively, buttons can be assigned to fulfill the roles of the links in the HTML Content.

Questions? Please email Section508mobile@va.gov. We're standing by to assist you, discuss preliminary content reviews early in the Development Process and provide some preemptive guidance to bring your Mobile Content into Section 508 Conformance.

ARIA's Hot Rolls (Roles)

We're seeing more and more content which contains ARIA roles these days. What is ARIA? WAI-ARIA is the Accessible Rich Internet Applications specification from the Web Accessibility Initiative at the W3C. ARIA elements



were developed in an attempt to improve accessibility of rich Internet applications (RIAs), providing the semantics to describe the role, state, and functionality of many familiar user interface controls, such as menus, sliders, trees and dialogs, as well as providing additional structural information to help developers identify landmarks, regions, and grids on their pages. Sounds great, doesn't it? So what's the problem?

First, the implementation of WAI-ARIA features, and support for these features, will vary among browsers. Also, most WAI-ARIA elements address only screen reader accessibility. Since other kinds of AT are not addressed, using these elements won't guarantee that ARIA is Section 508 compliant.

One ARIA role that can cause problems for screen reader users is `role=application`. When this role is used, the screen reader treats the content like an application rather than a web page or document. Once inside a web application, all keystrokes are passed through to the application. So that a screen reader user loses the ability to navigate using standard HTML navigation keystrokes.

Before you use `role=application`, be sure you're using it correctly. Ask yourself if your content is truly an application, or simply meant to be read as a document.

You can learn more about WAI/ARIA on [the W3C site](#), or on the [Section 508 training resources page](#)*

Is That Content Really Hidden?

CSS, JavaScript, and other DHTML techniques are sometimes used to hide or collapse on-screen content such as a set of links, another web page or a piece of text. Even though this content is intended to be hidden, screen readers may announce it to the user if the content is not marked as "hidden" in an appropriate way. When hidden text is announced, users may get the wrong information, or they may activate links that are not supposed to be available to them. This can lead to a very confusing and frustrating experience.

To prevent this problem, developers should set the `display` property to `none` and the `visibility` to `hidden`. This approach is more effective than moving the content off the screen or hiding it with a non-visible `z-order` position because using these properties will successfully hide the content from a variety of screen readers.

For assistive technologies supported by ARIA, the `ARIA-hidden` attribute can be set to `true` to hide content from screen readers. (See the "Hot Roles" article in this newsletter for more information about ARIA).

Relying on the HTML5 `hidden` attribute to hide page content is not recommended. Developers should use the CSS off-screen techniques shown here instead.

For more information about making content invisible to Assistive Technology, see the resources at [Web Accessibility Best Practices](#) and [Ensure hidden content is not rendered by assistive technology](#)*



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

NON-COMPLIANT EXAMPLE

```

<style type="text/css">
.class1 {
  position:absolute;top:-100;
}
.class2 {
  opacity:0;
}
.class3 {
  position:absolute;
  z-index:-99;
}
</style>

<div class="class1"> content not
displayed by browsers
</div>
<div class="class2"> content not
displayed by browsers
</div>
<div class="class3"> content not
displayed by browsers
</div>

<!--Non-compliant example of html5
hidden attribute -->

<div hidden ><p>HTML content should
not be rendered by the
assistive technologies such as screen
readers.</p></div>

```

COMPLIANT EXAMPLE

```

<div style="display:none;"> content
not displayed by browsers
</div>

<div style="visibility:hidden;">
content not displayed by browsers but
whitespace kept</div>

<div aria-hidden="true">content not
rendered by assistive technologies
that support the ARIA specification</
div>

<!-- Compliant example of html5 hidden
attribute -->

<div style="visibility:hidden;"><p>HTML
content should not be rendered by the
assistive technologies such as screen
readers.
</p></div>

```

**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: vaww.section508.va.gov*

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