Harmonized Processes for Section 508 Testing: **Baseline Tests for Software & Web Accessibility**

November 2016 | Version 2.0.1

Document status, review comments, and feedback

The current version is 2.0.1 approved for distribution by the Interagency Trusted Tester Program (ITTP) Technical Working Group.

For more information on the baseline tests and links to published streamlined test processes, training and certification programs, visit:

http://www.dhs.gov/compliance-test-processes

Contents:

Document status, review comments, and feedback 1
Introduction 4
Baseline tests 4
Background 4
How the baseline tests are structured 6
Use of the baseline tests by federal agencies and other groups 10
Agency issues beyond the test process 11
Developing a streamlined test process from this baseline—a primer 13
Examine example test processes first 13
Examine the advisory notes on each baseline test 13
Target audiences, requirement and test instruction wording 13
Test Process requirements 14
Modifications to the baseline tests 14
Test tool instructions 15
Reporting results 16
***The Baseline Tests (#1 - #28)* 17**
Attachment A - Cross-Reference Tables 80
Baseline tests (cross-reference table) 81
Section 508 (cross-reference table) 84
WCAG 2.0 (cross-reference table) 87
Attachment B - Flashing content test advisory notes 90
Why to include a flashing content test in a test process 90
Why there is no baseline test for flashing 90
Requirement, and draft rationale 91
How to report on flashing content 91
Attachment C - Baseline Test Report Checklists 93
Software-only test 94
Web-only test 95
Web+Software test 96
Summary of failures 98
Document Content Change Log 100
Version 1.0.6, March 2015 100
Version 1.1, February 2016 100
Version 2.0, October 2016 100
Version 2.0.1, November 2016 101

Contents: The Baseline Tests (#1 - #28)

1. Keyboard navigation 18
2. Focus (visible) 21
3. Focus (order) 23
4. Focus (Revealing hidden content) 25
5. Repetitive Content 27
6. Multi-state components 29
7. Images 32
8. Color (meaning) 36
9. Color (contrast) 38
10. *Flashing (reserved)* 40
11. Forms (associated instructions) 41
12. Page Titles 45
13. Data Tables (headers) 46
14. Data Tables (cell-header association) 49
15. Headings 52
16. Links and User controls 54
17. Language 57
18. Audio (transcripts) 59
19. Video (descriptions) 61
20. Synchronized media (captions) 63
21. Synchronized media (descriptions) 65
22. Style-sheet non-dependence 67
23. Frames 69
24. Alternate pages 70
25. Time outs 72
26. Image maps 74
27. Plug-in Links 75
28. Built-in accessibility features 77

Introduction

Baseline tests

This document contains accessibility design requirements and validation tests for software and Web accessibility. These tests are for measuring compliance with Section 508 of the Rehabilitation Act of 1973, as amended (29 U.S.C. 794d)[1].

This publication contains *baseline* tests that have been agreed upon and adopted by representatives of federal agencies as part of an effort to harmonize their testing processes for Section 508.

These baseline tests establish the minimum steps required to determine whether an application passes or fails applicable Section 508 technical and functional performance requirements. Federal agencies are encouraged to adopt the baseline to establish a consistent, shared, government-wide test approach.

Federal agencies and other groups are at liberty to develop their own test processes, incorporating the baseline tests and any additional test criteria specific to their needs. Although agencies may add some unique tests in their processes, all agency test processes will include the *baseline* test elements. A unified approach for 508 testing will provide consistency across government and confidence in test results shared between agencies.

Background

When Section 508 came into effect in 2001, it was up to individual federal agencies to develop their own responses to the requirements (interpretations of standards, development of test and governance processes, etc.). Some agencies do not test as part of their compliance efforts, lacking resources and/or expertise. Instead, they may rely on vendor claims of conformance with the Section 508 standards. A number of agencies do test, however, and have developed their own processes for evaluating Commercial-Off-The-Shelf (COTS) products, and internally developed software applications and Web sites. Because each agency developed its own test processes independently, there are inevitable differences in testing approaches and associated compliance determinations. Such differences have resulted in different testing outcomes for the same products: agency "a" would test COTS product "x", finding it 508 compliant, then agency "b" would find product "x" non-compliant. Without consistent testing, vendors receive mixed or conflicting messages from different federal agencies on the compliance of their products and services, and multiple agencies tend to test the same products due to a lack of trust with one another's test results.

In an effort to improve Section 508 testing across government, the "Harmonized Testing Process for Section 508 Compliance: Baseline Tests for Software and Web Accessibility" was developed as part of a collaborative project between accessibility teams at the US Department of Homeland Security (DHS) and the US Social Security Administration (SSA).

Prior to this project, SSA and DHS evaluated software and Web accessibility against Section 508 requirements using very different approaches. DHS evaluated against the requirements

---

[1] Section 508 is an act that requires all federal departments and agencies to ensure that their electronic information & technology (EIT) is accessible to people with disabilities. The specific standards for compliance with Section 508 are published at Section508.gov.

using code inspection tools to examine source code without the use of Assistive Technology (AT). SSA, on the other hand, used an AT-intensive approach, testing Web sites with assistive technologies most commonly used by SSA employees with disabilities. While the two approaches were radically different, they both had one thing in common: each strived to accurately evaluate software and Web accessibility for Section 508 compliance. Consolidating and harmonizing the two approaches was not easy, but for many reasons DHS and SSA felt it was necessary.

The result of this project is an agreed upon, harmonized core or *'baseline'* set of tests that agencies can use to develop their own test processes. Test processes should incorporate the baseline as the minimum, and agencies have the option to streamline/enhance their testing processes to include more than the baseline if needed (Figure 1).

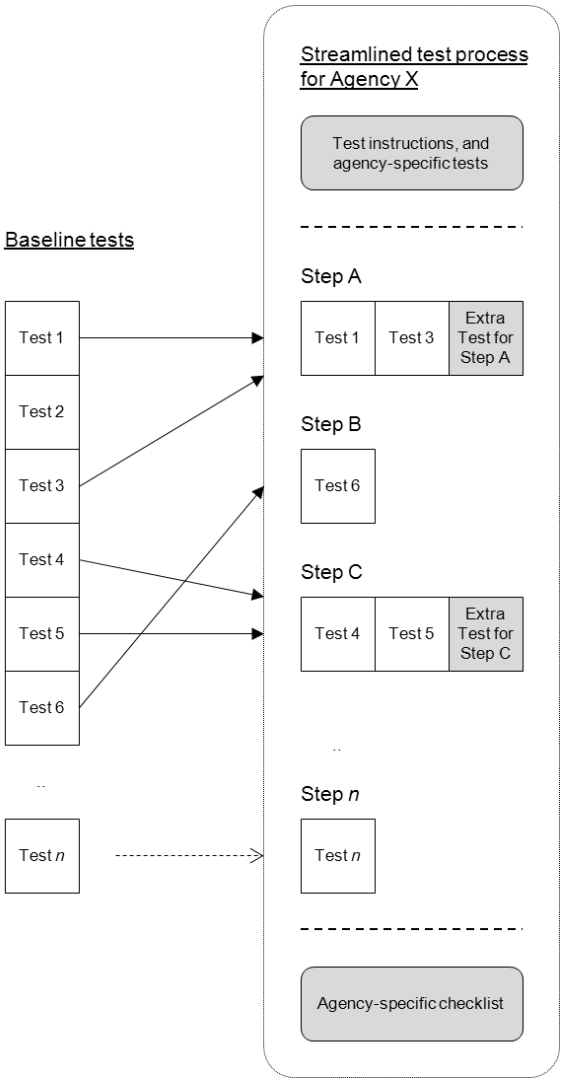

Figure 1 - Developing a streamlined test process incorporating the baseline tests

The baseline tests in this document have been developed to cover Section 508 technical requirements for Software, Web, and Functional Performance Criteria (FPC) that apply to all

Electronic and Information Technology (E&IT).[2] The government's Section 508 standards are, at the time of writing, under a revision process. While not definite, it is likely that the revised Section 508 standards will follow closely the World Wide Web Consortium (W3C) Web Content Accessibility Guidelines 2.0 (WCAG 2.0).[3] It was therefore decided to develop the 508 baseline tests to include, or at least align with[4] most of the WCAG 2.0 success criteria, in preparation for the upcoming Section 508 refresh. A cross reference to show how the baseline tests map to Section 508 and to WCAG 2.0 in Attachment A.

Given the trend for technologies to converge, WCAG 2.0 requirements were developed to accommodate inevitable future technology changes, and the language used in those guidelines is necessarily at a high level. The experiences testers at both SSA and DHS had, following WCAG 2.0 to the letter, were problematic. Some tests led to inconclusive results, and some were considered too subjective in nature. The development of the new baseline tests is the result of an attempt to reduce ambiguity, increase consistency of results, and emphasize the methods and techniques that can reliably meet the Section 508 requirements, given the current state and compatibility of underlying technologies.[5]

The result of the collaboration between SSA and DHS is reflected in the current document: a set of baseline tests that cover the current Section 508 standards, that align with applicable WCAG 2.0 Level AA success criteria, and that can be incorporated in separate, practical, systematic test processes for software application and Web accessibility.[6] Additional WCAG harmonization may be explored in the future as the Section 508 refresh process continues and testing tools and techniques mature.

How the baseline tests are structured

The selection criteria for including requirements and baseline tests against those requirements was:

- **Standards based:** The requirement must be firmly rooted in standards (both current and emerging), or is there to address specific, documented, high-risk accessibility issue (complaints are documented in an area that the standards did not anticipate).
- **Validated:** Tests were validated by SSA and DHS, and are known to produce reliable and repeatable results. In future updates, validation tests must be conducted by more than one agency.
- **Usable:** Validated Baseline tests were adapted into a practical formal test process that was tested to verify usability.

The tests have been developed to contain sufficient information and instruction to make a consistent and unambiguous measurement of the accessibility of interface components, independently of the other tests. However, doing all of these tests in sequence is not

---

[2] E&IT is more widely referred to as ICT (Information and Communications Technologies).

[3] Web Content Accessibility Guidelines (WCAG) 2.0, W3C Recommendation 11 December 2008. Available: http://www.w3.org/TR/WCAG20/

[4] Note that "aligns with" does not imply "conforms to". For conformance with WCAG 2.0, a WCAG 2.0 test process should be followed, rather than the Baseline.

[5] For example, HTML (Hyper Text Markup Language), ARIA (Accessible Rich Internet Applications), platform APIs (Application Programming Interfaces), browsers, and assistive technologies.

[6] The baseline tests herein are aligned with the WCAG Level A and Level AA success criteria. WCAG comments on the more stringent AAA: *"It is not recommended that Level AAA conformance be required as a general policy for entire sites because it is not possible to satisfy all Level AAA Success Criteria for some content."* http://www.w3.org/TR/WCAG20/#cc1

recommended. Instead, testers should follow a streamlined, practical test process that incorporates these baseline tests (Figure 1).

Platform, browser and tools

The baseline tests support the following browsers and operating systems.

- Microsoft Internet Explorer version version 11
- Chrome (version 49.0.2623.87).
- Firefox (version 45.0.2)
- Microsoft Windows versions 7, 8.1, and 10. (In Windows 8.1 and 10, testing is performed in Desktop mode.)

It is recognized that product updates may take place daily, weekly, or monthly, and specific versions noted above may quickly become outdated. If a version update creates critical issues with usage within the baseline testing procedures, details will be provided and those versions shall be prohibited.

The tools used in the baseline tests have been chosen based on several factors including ease of use, ease of teaching, and accuracy of results. They are also free to use. Installation instructions for these tools are available at https://www.dhs.gov/dhs-section-508-compliance-testing-tools.

- **"Inspect"** from Microsoft Corporation. This tool reveals the accessibility properties (Name, Role, Value and State) of Windows software components.[7]
  - URL for Windows 7: http://www.microsoft.com/en-us/download/details.aspx?id=8279
  - URL for Windows 8.1: https://msdn.microsoft.com/en-us/windows/desktop/bg162891.aspx
  - URL for Windows 10: https://developer.microsoft.com/en-US/windows/downloads/windows-10-sdk
- **"Java Ferret"** from Oracle Corporation. This tool reveals the accessibility properties (Name, Role, Value and State) of Java software components.[8]
  - URL: http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136191.html
- **"Web Accessibility Toolbar (WAT)" versions 2012 or 2015 for IE** from the Web Accessibility Tools Consortium. This adds a toolbar to Internet Explorer to aid manual inspection of accessibility properties of components on web pages. **Note**: This toolbar is only compatible with IE. To test with Firefox or Chrome, install WAF instead (see below).
- URL: https://www.dhs.gov/dhs-section-508-compliance-testing-tools
- **Bookmarklets/Favelets.** These tools are JavaScript testing functions that are activated in the browser, and supplement WAT. (**Note**: These favelets are not necessary if using WAF.)
- URL: https://www.dhs.gov/dhs-section-508-compliance-testing-tools
  - **"ARIA Markup Favelet"** reveals ARIA attributes on web pages.
  - **"Named Anchors Bookmarklet"** reveals anchor tags on web pages

[7] Inspect is a component of the installation of Microsoft Windows SDK for Windows 7 and .NET Framework 4, Version 7.1, or Microsoft Windows SDK for Windows 8.1 and 10. The tool must be used in the User Interface Automation (UIA) mode, which includes the Microsoft Accessibility Architecture (MSAA) properties.

[8] Java Ferret is a component of the installation of Java Access Bridge 2.0.2.

    - **"Frames Favelet"** reveals frame and iframe properties on web pages
- **"Web Accessibility Favelets (WAF)"**. These favelets were developed as an alternative to WAT. They can be installed in IE, Firefox, and Chrome. WAF duplicates the functionality of WAT (except it includes the Bookmarklets/Favelets but not the Colour Contrast Analyzer) as much as possible within the limitations of the favelet security context. (See below for more details on known differences.) The Skip Link favelet from Jim Thatcher's site is included.
    - URL: https://www.dhs.gov/dhs-section-508-compliance-testing-tools
- **"Colour Contrast Analyzer"** from The Paciello Group. This tool is included with WAT. It can also be installed as a standalone executable (to supplement WAF).
    - URL: https://www.dhs.gov/dhs-section-508-compliance-testing-tools

While there are other platforms, browsers and tools available, those used herein have been technically validated for accurate and repeatable results. Agencies that use other technologies are encouraged to develop an equivalent process for their specific test environments. Contact the authors (see contact details at the front of this document) to verify use of those technologies. Once use and adjustments have been verified, results of modified product test processes may be shared. Agency-specific software installation and use guides should be included in streamlined test processes based on these tests.

Notes on Browser Differences

There are notable differences in test results depending on the test environment. Specifically, browsers (and browser versions) provide varying levels of support for certain code techniques. Browser differences may cause test results to vary for an application. Test results for the test environment should be reported (NC/ C/ NA) as instructed in each baseline test. It may be necessary to test an application in multiple test environments in performing root cause analysis of issues. These caveats will be repeated in this document where relevant.

The following differences have been identified:

1. Chrome accessibility mode: When testing in Chrome, ensure that accessibility mode is enabled by either:
    a. navigating to `chrome://accessibility` and setting this globally or per tab, or
    b. Starting Chrome with the `--force-renderer-accessibility` flag. (See https://www.chromium.org/for-testers/command-line-flags for how to start Chrome with command-line flags.)
2. Flash: In Chrome, the Name, Role, and State of Flash elements are not revealed through accessibility protocols. When Flash is tested in Chrome, expect the test results to be Not Compliant for most of the software requirements and keyboard access due to Chrome restrictions. While these are accurate test results in the Chrome environment, the test results for the Flash may differ if tested in another browser. Firefox may also affect the test results for Flash. To determine the accessibility of the coded Flash content, test the application in IE11.
3. Java: Support for Java in Chrome and Firefox has ended or will end. A tester may not be aware of unsupported Java content in Chrome since there is no indication of missing content. Expect the test results for Java in these browsers to be Not Applicable or Not Compliant due to lack of browser support. To determine the accessibility of the coded Java content, test the application in IE11.
4. ARIA forms: Differences in test results between browsers were identified for forms that utilize ARIA for labelling. To determine the accessibility of the coded ARIA forms, test the forms in Firefox or Chrome.

Notes on WAF

The baseline tests were originally developed using WAT, which is ony compatible with IE. To include other browsers in the approved test environment and minimize changes to test instructions, WAF was developed to replicate WAT operation and functionality as closely as possible. Alternative instructions are not provided for WAF unless there is a significant difference from WAT in terms of testing or interpreting results.

WAF is a set of favelets that can be easily installed on various browsers. The functions of the WAF favelets were named according to their corresponding functions in WAT, minus the menu hierarchy, and the instructions for testing can be easily mapped between WAT and WAF. For example, where the WAT instruction is

> Use the **WAT (Doc Info - ShowTitles, Images - Show Images)** to examine …

If using WAF, interpret these instructions as

> Use the **WAF (Show Titles, Show Images)** to examine …

Once installed, the tester can organize the favelets by any preference (e.g. in alphabetical order, in subfolders, in testing order, etc.).

WAF nuances: .

- Disabling CSS: Whereas **WAT (IE - Toggle CSS)** disables or enables CSS until this function is re-selected, **WAF (Toggle CSS)** only disables CSS for one page refresh. Additionally, **WAF (Toggle CSS)** may not re-enable CSS completely, in which case you a page refresh will be required.

These caveats will be repeated in this document where relevant.

Baseline Tests

There are 28 separate requirements with associated tests, covering all relevant components for software applications and Web sites. Each test contains the following information:

- **Numbered Requirement:** In plain English, how the component(s) should function in order to meet the related standards. Note that the numbers are arbitrary, and do not infer a practical test sequence
- **Rationale:** In plain English, an explanation of why this test is important, and why the test methods are appropriate, with particular regard to the type(s) of disability-related problems being addressed.
- **Related Standards:** Which of the Section 508 standards are addressed by this test. Also, which of the relevant WCAG 2.0 success criteria this test aligns with. A given 508 standard or WCAG criteria may be addressed by multiple baseline tests.[9]
- **Tools Necessary:** Testing technologies used in this test.
- **Test Instruction 1 - Finding Applicable Components:** How a tester would find the components that need to be tested.
- **Test Instruction 2 - Inspecting/Using Components:** How a tester would determine whether the components found in Instruction 1 meet the requirement. This is achieved using inspection tools, and using human judgment.
- **Test Instruction 3 - Failure conditions:** A list of possible outcomes from Instruction 2, along with what to mark on a test report for this particular test.
  - **3a - Section 508 Failure Conditions:** The technical requirements and/or functional performance criteria that should be marked as failures in test results. Only failure conditions are given for Section 508.[10]
  - **3b - WCAG2 Failure Conditions:** The A or AA criteria that should be marked as failures in test results.
  - **3c - Baseline Requirement Test Results:** This includes a complete list of conditions that must be fulfilled in order to pass the baseline requirement, and conditions under which the baseline requirement is not applicable. Note that any failure in 3a means that the baseline requirement fails.[11]

Appended to each test are an advisory notes entitled "**Tips for streamlined test processes".** These tips provide additional information, specific to the current test, that support the generic information provided in the primer section of this document.

Use of the baseline tests by federal agencies and other groups

Federal agencies and other groups are encouraged to adopt these baseline tests, and either develop their own test processes, or follow a test process developed by another agency.[12]

When developing test processes, and reporting results from such test processes, agencies must take note of the following:

- Test **results for each baseline requirement must be reported**. As such. each baseline requirement must be incorporated into the test process to be considered an acceptable test process.

---

[9] Cross-reference tables are provided at the back of this document.

[10] Streamlined test processes may include statements of when to mark Section 508 standards as compliant, or as not applicable.

[11] For sharing test results between agencies, the results of 3a and 3c must be reported.

[12] The baseline tests should not be re-published without citation, nor should they be modified from the content herein, which has been agreed upon and adopted by several government agencies.

- Results of tests that **incorporate the baseline tests** are considered repeatable. A conclusion as the result of such a process can say that it is on the agreed-upon baseline.
- Test processes that **do not include all baseline requirements** are not considered to be following the baseline. Results of these test processes will not be accepted by agencies that have adopted the baseline tests..
- Results of tests that incorporate the baseline but also **go beyond the baseline** with additional test requirements (see also Figure 1), must clearly separate out in the report the results that refer to the baseline, and the results that refer to additional agency-specific tests.

Agency issues beyond the test process

It should be noted that use of a test process that incorporates the baseline tests is affected by other contextual issues that accompany any Section 508 program in a federal agency. Some examples of related issues to consider are:

- This document does not address the policies or organizational disciplines necessary to develop a Section 508 program or outline the processes needed for acceptance of vendor deliverables.
- Our goal is to clearly document the accessibility of the evaluated content against the Section 508 technical and functional performance requirements that are applicable to software and web. While the baseline provides a more predictable and reliable way of evaluating content, the test results can be regarded as one factor that goes into making a Section 508 compliance determination (the choice of an agency to adopt an application or not). Other factors to consider in making compliance determinations include, but are not limited to legal issues related to acquisition[13], technical issues of compatibility with existing systems, and business needs. The output of a test process incorporating the baseline test will provide results that can assist in making compliance determinations and acceptance decisions of contract deliverables. The results may also be used to notify vendors and teams of defects, and plan for / prioritize ongoing test and remediation tasks.
- This document does not address how to handle coding mistakes. Problems may be found during testing that impact accessibility, but are simply coding errors. Included here would be things like links that lead to the wrong target website. A tester may be responsible for notifying a developer if that is agency good practice, but these issues are usually not considered Baseline test results.
- The baseline test methodology does not include tests with assistive technology. Agencies must decide the role assistive technology will play in their internal testing program and Section 508 compliance determinations. Compatibility and usability of content with assistive technology plays an important role in assuring people with disabilities have comparable access to technology, information, and systems.[14] Because AT testing can result in false-positives and false-negatives, defects must always be confirmed with the baseline methods herein. Additional testing with AT may reveal conclusive insights, but caution is urged: experience shows that such additional AT testing is proficient only with experienced, well-trained testers.
- Section 508 puts Federal Agencies and some entities that receive federal funds at both business and legal risk if they do not comply with the law. The developers of the baseline (at DHS and SSA) recognize that a well implemented Section 508 program manages risks and

[13] Federal Acquisition Regulation (FAR 39.2) https://www.acquisition.gov/far/html/Subpart%2039_2.html

[14] Some content (e.g. dynamically generated content such as use of AJAX and ARIA) may produce passes and fails, the impact of which cannot be determined without testing with assistive technology.

knowingly takes on some risks. For example, an agency may deem it acceptable to use this baseline to document minor deficiencies and allow content to be published and applications to be pushed to production. However, when evaluating COTS products, the severity of the impact of a given defect or set of defects should be up to the implementing agency (and not another agency or vendor). If results are generated outside of the implementing agency (e.g. another trusted agency or vendors), they should ignore any severity levels. In summary, agencies should not accept outside entities evaluating their exposure to risk.

- The baseline test methodology does not include guidance on managing a testing program. Agencies must determine the rules and procedures that their testers will follow in performing testing to ensure adequate testing of applications. This includes when testing will be performed, at what level of coverage, and in which test environment. As noted in the Browser Differences section, the test environments may affect the test results of some applications. Agencies should consider these factors in determining standard operating procedures for their testing program.

Developing a streamlined test process from this baseline—a primer

The baseline test tables in this document are not intended to be followed in a linear fashion, and should be enhanced to form streamlined test processes for given audiences (see also Figure 1, page 5). The following notes give a primer on issues to consider while developing a streamlined testing process.

Examine example test processes first

Other federal groups have developed streamlined test processes. The work you are planning may already have been done. Agencies publishing their test processes usually allow other agencies to adopt and use them.[15]

Examine the advisory notes on each baseline test

Each baseline test table in this document has a row entitled "Advisory: Tips for streamlined test processes". These are tips on how tests may be combined, how tests might easily be enhanced, and so forth. These notes should always be consulted when creating a test process, although they are advisory in nature.

Target audiences, requirement and test instruction wording

The baseline tests have been produced with the assumption that testers have training / skills in accessibility, and have a basic understanding of HTML and the construction of Web pages. Testers must also have knowledge of the content or application that they are testing, or they must be able to follow an informed test plan.

It is also assumed that testers have necessary skills to evaluate subjective information in context (e.g., the suitability of alternate text for images). Any agency adopting the baseline tests and producing their own streamlined process (or adopting a published process) must ensure that testers are given proper documentation, test plans, demonstrations, and access to developers for clarifications and explanations, as appropriate. Any test process incorporating these baseline tests must therefore be tailored to the specific needs of its developers and/or testers.

The baseline tests could be written for an audience of developers, an audience of testers, or an audience of both. The requirements in each of the baseline tables have been presented in a neutral tone that is component-specific (e.g., "Links and/or user controls must have meaningful names"). It may be desirable to reword the requirements and instructions targeting developers (e.g., "Provide meaningful names for all links and/or user controls"). Alternatively, it may be that the process will be used only by testers, and so the language might be changed to reflect that (e.g., "Check that links and/or user controls have meaningful names").

[15] Note any copying or editing restrictions etc. given in each published process.

Test Process requirements

The test process contains all instructions that a tester needs to follow the process completely to test a product and report on the product's test results. Test processes derived from the baseline tests should include a mapping to each baseline test and each Section 508 requirement. Test processes should include the following:

1. Testing Tools
    a. Where to obtain testing tools
    b. How to set up of tools to ensure consistent test results between testers
    c. How to use the tools
    d. Non-baseline tools must be identified and results from these tools are not to be used to determine baseline test results.
2. Testing instructions
    a. All baseline tests must be included (including an agency method for Test #10 Flashing)
    b. Advisory tips for streamlined test processes may be incorporated.
    c. Agency-specific, non-baseline tests must be identified and not affect baseline test results
    d. Test instructions (methods and use of tools for testing)
    e. Define failure conditions
    f. Define the 508 standard(s) and Baseline Tests that are being tested

Modifications to the baseline tests

Given the nature of the baseline tests, they are not intended to be used for testing 'as-is'. Creation of a streamlined test requires some amount of modification to the baseline. The following provides guidance on what to do, and what not to do when modifying the baseline content.

Test order

Baseline tests included in this document are not intended to be used in a linear fashion. The order with which tests are conducted may be changed from the order herein (the numbers of each baseline test are for reference only). Tests may be combined for efficiency. For example, keyboard and focus tests can usually be done at the same time.

Always include the baseline, enhance as needed

Agencies that adopt the baseline tests agree to always incorporate each baseline test listed herein in their streamlined test processes. "Modification" in this sense does not allow for dropping *any* of the baseline tests.

To adopt the baseline, the content in each of the following table cells of the baseline must be represented somewhere in the streamlined test process (as a minimum):

- Numbered Requirement
- Rationale
- Test Instruction 1 - Finding Applicable Components
- Test Instruction 2 - Inspecting/Using Components
- Test Instruction 3a - Section 508 Failure Conditions
- Test instruction 3c - Baseline Requirement Test Results

Additional agency-specific tests (see below) must be identified as agency-specific testing (for example, by means of a cross-reference table appended to the test process document).

Wording changes, yes; Meaning changes, no

It may be desirable to change wording from the baseline. For example. it may be desirable to change passive voice to active voice. Wording changes to create a smooth-flowing, easy to read document are acceptable. However, care should be taken to ensure that the meaning remains the same even though the words used are different. A reviewer of any streamlined test should be able to compare the content to the baseline and conclude that the meaning and results that would come from a test remain the same. Any errors or suggested improvements to the baseline should be submitted to the address on page 1 of this document.

Separating out seldom used information

Each baseline test contains a rationale and a list of the necessary tools. Testers need to learn this sort of information once, and then have it available for quick reference. It is perfectly acceptable to separate such seldom-used information into a separate section, but this information must stay with any published test process (or be available to access from any online streamlined test tool incorporating these baseline tests).

Additional agency tests beyond the baseline

Agencies have the option to enhance their test processes to include more than the baseline if needed (see also Figure 1 on page 5). For example, there is a test in the baseline that headings, where used, are programmatically marked up so that they are accessible to screen reader users. Any agency may decide to create a policy that "reports and memos over 1500 words long must include headings, to enhance readability and enhance accessibility". In this case, the test becomes (a) whether headings exist to break up text over 1500 words long, and (b) whether existing headings programmatically marked. When it comes to creating such a streamlined test, and when it comes to sharing the results of such a test between agencies, the agency-specific test (a) should be omitted (or at least clearly marked as a non-baseline test); and the baseline test (b) should always be included in the same manner as for the other baseline results.

An agency may also create a streamlined process that includes guidance to their accessibility test teams and other personnel on when a given baseline test failure does not result in an agency compliance determination failure. As stated earlier (see *Agency issues beyond the test process*), test results are only one factor in making internal compliance determinations. An agency policy that accepts a certain baseline failure is the decision of that agency only. While the streamlined process may include such information and guidance for an agency's own internal use, results from it should similarly be separated out from reports when sharing baseline test results between agencies (in other words, report against the baseline; not against the compliance determination).

Test tool instructions

Each baseline test lists the tool(s) used in that test. The test instructions provide the high-level instruction on which part of the tool to use (normally a menu choice). Instructions on how to use each testing tool are not included in this document, but should be provided to testers, either as part of a streamlined test process, or its accompanying documentation.

It may be useful to visually differentiate test results, HTML and other code in the streamlined process. For example:

In the Baseline:

> Use WAT (Tables – Show Data Tables). Each row and column header must have either a SCOPE="col/row"; or a ID="x". If ID is used, data cells must refer to the associated header cell's ID in order for the header to pass this test.

In a streamlined process:

> **WAT** > Tables > Show Data Tables.
> - Each row and column header must have either
  - `SCOPE="col/row"`; or
  - `ID="x"`.
- If `ID` is used, data cells must refer to the associated header cell's `ID` in order for the header to pass this test.

Reporting results

Each baseline test includes an instruction (#3) for test results. Results are presented in terms of a clause, followed by, (3a) failures of Section 508, (3b) Failures of WCAG 2.0, and (3c) failures or passes of the Baseline Requirements. The results of 3a and 3c must be reported, and 3b may optionally be reported.

The method used in the baseline is to give certain clauses and then the standard, guideline or requirement that is impacted by that clause. An agency developing a streamlined test process can present failures in a way that meets their testing needs. For example, a clause and failure is given in the baseline as:

[Web only] The purpose and/or function of a non-decorative image is not properly conveyed in descriptive text (1194.22(a): Equivalent text descriptions)

This could be written in a streamlined process as:

[W]	Purpose and/or function of non-decorative image not properly conveyed in descriptive text. Fail 22a

Failures must be explained in the report. Reports will generally include things like the type of failure, the location of the failure, and supporting screen captures with test tool results. Reports may also describe any peer review process used.

When sharing reports between agencies, a checklist should be included. Checklists for software-only, web-only, and software plus web combined, are included at the back of this document (Attachment C).

It is not required that the compliance determinations[16] that follow on from test results be included in any test reports that are shared between agencies. Including such information is not discouraged, however.

[16] Compliance determination may be based on the test results, as well as many other applicable factors (see advice in *Agency issues beyond the test process*, page 9).

The Baseline Tests (#1 - #28)

1. Keyboard navigation

Requirement	1. Keyboard access and control must be available for all interactive interface components and frames that can be accessed or controlled with a mouse. Where non-standard keyboard commands are employed, notify users of the existence and use of Alternate keyboard commands through the interface, application help, and/or documentation.
Rationale	Interactive interface components include navigation controls (links, buttons etc.), and editable content (selectable text, data input etc.). Wherever users are expected to interact with components, it must be possible for users to get to those components or perform those functions using only the keyboard, because (i) using a mouse is not possible when the user has no sight, and (ii) using a mouse is not possible when the user does not have the physical capability / dexterity to effectively control a pointing device. Keyboard access is defined as use with physical keyboard that is attached to the computer, either separately (desktop PC) or integrated (laptop PC, kiosk). Ideally, interfaces use standard keyboard commands (TAB, Space Bar, Enter, Escape, etc.), making their use easy and efficient. On occasions, an interface may be designed to expand on the basic set of standard keyboard commands; and/or remap standard keys. In both of these cases, users must learn the non-standard keys. In order to be aware of non-standard key commands, users must be notified of their existence and correct use. **Notes:** • Access must be via a physical keyboard. Specifically excluded from this test is the use of an on-screen keyboard, or using the Mouse-Keys feature in Windows. • At this time the baseline tests herein cover use of software and Web sites on PCs (i.e., desktops and laptops) that have a keyboard as a primary input device. Tablet PCs and software running on other portable devices are not addressed in the baseline tests.
Related Standards	508 1194.21(a): Keyboard Accessibility WCAG2: 2.1.1 Keyboard WCAG2: 2.1.2 No Keyboard Trap WCAG2: 1.3.1 Info and relationships
Tools Necessary	Physical system keyboard and pointing device (e.g., mouse), WAT
Test Instruction 1: Finding Applicable Components	a. Find all visible and hidden interactive interface components (links, form fields, drop down menus, show/hide content, tree views, pop ups/light boxes, frames, iframes, etc.) using a mouse (hover and/or click). b. Use WAT (Doc Info-Show Titles) to reveal information that will be revealed by mouseover through the TITLE attribute.

Test Instruction 2: Inspecting/Using Components	a. Use the standard keyboard commands (Tab, [Shift+Tab], Space bar, ALT, arrow keys, Enter, etc.) to navigate through each interactive interface component (including form drop-down lists and form fields), reveal hidden content, and activate all interface components. b. Inspect any help (contextual help, or application help) and documentation for notification of available Alternate keyboard commands (e.g., non-standard keyboard controls for all users, or access keys, hotkeys). Where standard keyboard commands do not work, there must be instructions for (i) extending standard keyboard command operations (e.g., getting out of a keyboard "trap"), and/or (ii) remapped/ alternate keys. Verify that non-standard keyboard commands they can be used to address deficiencies found in step a. c. Inspect the information provided through the TITLE attribute on all interactive components (including images that are interactive). If the TITLE information in the TITLE attribute cannot be revealed by keyboard, it must be conveyed through screen text or visual context. **Notes:** • If a "trap" disrupts keyboard navigation, note the failure and use a mouse to regain control beyond the trap to continue testing. • The test of whether an interactive interface component cannot be accessed and/or activated by the keyboard can be satisfied by either step a, or by step b . • Non-standard keyboard access methods or shortcut keys must be documented in a help section or be apparent on the screen (hotkeys become visible when pressing the Alt key, underlined letter, etc.). • **[Web only]** Skip link keyboard navigability is a part of this test. • **[Web only]** If using the keyboard reveals the TITLE attribute's information (e.g., through scripts), then it is not necessary to have that information on the page. • Interactive Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• An interactive interface component or function cannot be accessed by the keyboard.
	  ○ Fails 1194.21(a): Keyboard Accessibility.
	• An interactive interface component or function cannot be activated by the keyboard.
	  ○ Fails 1194.21(a): Keyboard Accessibility.
	• A "trap" disrupts keyboard navigation.
	  ○ Fails 1194.21(a): Keyboard Accessibility.
	• Information provided by the TITLE attribute is not revealed by the keyboard and is not permanently shown on screen
	  ○ Fails 1194.21(a): Keyboard Accessibility.
	• Interactive interface components and functions can be accessed AND/OR activated by the keyboard BUT non-standard/Alternative commands are undocumented.
	  ○ Fails 1194.21(a): Keyboard Accessibility.

Test Instruction 3b: WCAG2 Failure Conditions	• An interactive interface component or function cannot be accessed by the keyboard.
	  ○ Fails 2.1.1 Keyboard
	• An interactive interface component or function cannot be activated by the keyboard.
	  ○ Fails 2.1.1 Keyboard
	• A "trap" disrupts keyboard navigation.
	  ○ Fails 2.1.2 No Keyboard Trap
	• Information provided by the TITLE attribute is not revealed by the keyboard and is not permanently shown on screen
	  ○ Fails 2.1.1 Keyboard
	• Interactive interface components and functions can be accessed AND/OR activated by the keyboard BUT non-standard/Alternative commands are undocumented.
	  ○ Fails 1.3.1 Info and relationships

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #1
	• All interactive interface components and functions can be accessed AND activated by the keyboard, AND any non-standard/ Alternative commands are documented.
	  ○ Passes Baseline Requirement #1

Advisory: Tips for streamlined test processes	• This test may be combined with tests for focus. • It may be useful to separate out a test for keyboard use, and then have a separate test for documentation of non-standard commands. • Tips and techniques for finding hidden content may be useful for testers. • It may be useful to provide a Windows keyboard reference guide to testers.

2. Focus (visible)

Requirement	2. There must be a visible indication of the currently focused interactive component.
Rationale	Some software applications, and Web browsers by default, indicate focus, but this can be disrupted by the application of custom programming, styles, style sheets, and scripting. However, such programming can also be used to enhance visual indications of focus to help users who have low vision. When controlling the interface with keyboard only, if there is no visual differentiation between the current focused item and the rest of the interface / content, then it is not possible to tell where in the interface you are. Therefore, a visual indication of focus is necessary.
Related Standards	508 1194.21(c): Visual Focus WCAG2: 2.4.7 Focus Visible
Tools Necessary	Physical system keyboard
Test Instruction 1: Finding Applicable Components	a. Find all visible and hidden interactive interface components (links, form fields, drop down menus, show/hide content, tree views, pop ups/light boxes, etc.) using a mouse (hover and/or click).
Test Instruction 2: Inspecting/Using Components	a. Using the keyboard, navigate to each interactive component and look for a visible indication of focus (usually an outline around the component). **Notes:** • The clarity of visible focus is subjective and the minimum level is the browser's (or OS platform) default display setting for indicating focus. • Some components that are not normally considered interactive may actually be in the tab order, and therefore interactive (e.g., screen text for form filling instructions). Such components should receive a visible indication of focus when tabbed to. • **[Web only]** Skip link visual focus is a part of this test. • **[Web only]** Loss of focus should not occur while manually shifting focus through the page (using the TAB or arrow keys). However, when a function that moves the focus is executed (such as an internal page link or hidden content is revealed), it may be necessary to manually shift focus once with the keyboard before focus becomes visible again. This is not considered a failure. • Interactive Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• An interface component does not give a visible indication when it receives focus.
	  ○ Fails 1194.21(c): Visual Focus.
	• A visual indication of focus occurs somewhere other than on the component that has focus
	  ○ Fails 1194.21(c): Visual Focus.

Test Instruction 3b: WCAG2 Failure Conditions	• An interface component does not give a visible indication when it receives focus.
	  ○ Fails 2.4.7 Focus Visible.
	• A visual indication of focus occurs somewhere other than on the component that has focus
	  ○ Fails 2.4.7 Focus Visible

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #2
	• All interface components give a visible indication when they receive focus.
	  ○ Passes Baseline Requirement #2

Advisory: Tips for streamlined test processes	• Explain how to determine the browser's (or OS platform) default behavior for indicating focus.

3. Focus (order)

Requirement	3. When the sequence of interface components has meaning or requires an order of operation, the focus/TAB order must logically flow with the application/content.
Rationale	A logical order and groupings of interface components is normally a given in the design of software applications and Web content. Groupings and order are usually visually apparent. Logical arrangements are used to aid visual appeal and improve usability. However, when the focus/TAB order does not follow the logical order, users can become confused, make errors, and may not understand the contextual meaning of components. This is especially true for people who have no vision, or who have low vision, and are relying on AT.
Related Standards	508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 2.4.3 Focus Order WCAG2: 3.2.3 Consistent Navigation
Tools Necessary	Physical system keyboard
Test Instruction 1: Finding Applicable Components	a. Examine the interface to determine the groupings and logical order. b. Find components that repeat on multiple pages or software screens (e.g., navigation menus).
Test Instruction 2: Inspecting/Using Components	a. Use the keyboard to navigate through the components. Be careful to address any hidden content. Note any instances where the order deviates from logical groupings, and logical order between individual components. b. Where components are repeated on multiple pages or software screens, note any changes to the relative order of the repeated components. **Notes:** • Interactive Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• There are mismatches between the TAB order and the logical order.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.
	• The relative order of repeated components changes between pages / software screens .
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• There are mismatches between the TAB order and the logical order.
	  ○ Fails 2.4.3 Focus Order
	• The relative order of repeated components changes between pages / software screens .
	  ○ Fails 2.4.3 Focus Order
	  ○ Fails 3.2.3 Consistent Navigation

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #3
	• The TAB order matches the logical order and the order of repeated components remains constant between pages.
	  ○ Passes Baseline Requirement #3

Advisory: Tips for streamlined test processes	• This test is for interactive interface components, excluding forms which are covered by the forms test. • To get to all components, it may require more than simply TABbing between items. For example, it may be necessary to tab to a set of components then use the arrow keys to get focus on individual components. • Tab order may be application specific—reflecting business logic—so it may be helpful to ask developers whether a seemingly non-logical order was intentional. It may be useful to verify order discrepancies using the Tab Index attribute, if it is present (Although a Tab Index is not required). A Tab Index Favelet is available from the same source as the ARIA Markup Favelet. It is also possible to Tab through components to see if there is a visual focus on static text. • For web content that is in layout tables, it is possible to produce a linearized representation that may be useful in determining whether a logical order is used. To linearize table content, use WAT (Tables - Linearize).

4. Focus (Revealing hidden content)

Requirement	4. Components that reveal hidden content (dialog boxes, light boxes, pop-ups, content accordions, drop-down menus etc.) must either (i) shift focus to the content they reveal, or (ii) the component must describe that a change to the content will occur if selected.
Rationale	Some components on web content and software screens are intentionally hidden to reduce visual clutter. Other components only appear as part of a procedure, such as an error notification . It is possible to reveal content with interface components in an inaccessible manner, by requiring user vision and/or requiring the use of a mouse. Keyboard users need to be able to get to the information and controls that are revealed, and users without vision, or with low vision, need to know that new content has appeared.
Related Standards	508 1194.21(c): Visual Focus 508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 2.4.3 Focus Order WCAG2: 3.2.2 On Input
Tools Necessary	Physical system keyboard, WAT, Inspect, Java Ferret
Test Instruction 1: Finding Applicable Components	a. Find instances of interface components that reveal hidden content, such as dialog boxes, light boxes, pup-ups, content accordions, drop-down menus.
Test Instruction 2: Inspecting/Using Components	a. Move the focus to the control that reveals hidden content, activate the control with the keyboard, and then determine whether focus is in the revealed content. It may be necessary to TAB once to find the focus. Continue to move through the revealed content using the keyboard. b. If focus does not shift to the revealed content, an accurate description of the content change event must be provided. ○ **[Web only]** Use the WAT (Doc Info - Titles, Images - Show Images) to examine the control's name, title and any adjacent screen text or ALT text. ○ *[SW only]* Use Inspect/Java Ferret to examine the control's Name and description. **Note:** • Without exception, focus must shift to modal dialog boxes (must meet step a, above) and remain within the dialog box until the box is closed by the user. • Interactive Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• A modal dialog box does not receive focus when it is opened.
	  ○ Fails 1194.21(c): Visual Focus.
	• A modal dialog box allows focus to move off the dialog box before the box is closed by a user's actions.
	  ○ Fails 1194.21(c): Visual Focus.
	• Focus does not move to revealed content, and no description of the content change is provided.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• A modal dialog box does not receive focus when it is opened.
	  ○ Fails 2.4.3 Focus Order
	• A modal dialog box allows focus to move off the dialog box before the box is closed by a user's actions.
	  ○ Fails 2.4.3 Focus Order
	• Focus does not move to revealed content, or instructions/status are not provided when focus does not move to revealed content.
	  ○ Fails 2.4.3 Focus Order
	  ○ Fails 3.2.2 On Input

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #4
	• Focus moves to the revealed content, or instructions/status are provided when focus does not move to hidden content.
	  ○ Passes Baseline Requirement #4
	• There is no hidden content.
	  ○ Not applicable (Baseline Requirement #4)

Advisory: Tips for streamlined test processes	• It may be useful to remind testers that keyboard access and visible focus should be tested also during this test. • Instructions on what "modal dialog boxes" are and how they should behave should be included. • Instructions for the use of Inspect / Java Ferret for identifying focusable content should be included for testers.

5. Repetitive Content

Requirement	5. **[Web only]** A method must be provided to skip blocks of repeated content or links on pages, allowing a user to move directly to page-specific content.
Rationale	Groups of navigation links are usually provided along the top and/or left of multiple pages to provide quick navigation to other areas of a Web site. In addition, some groups of pages may repeat blocks of content (other than navigational controls). For users who can see and use a mouse, skipping over navigation links and other blocks of content is simply a mouse movement followed by a click. However, for users who cannot use a mouse, repetitive links can be a serious impediment to productivity. If a site has forty repetitive links, a keyboard user must complete forty keystrokes just to get to the information they need to use on each and every page. To enable equitable use by keyboard only users, there must be a method to skip past repetitive content. Similarly, for screen reader users, if they must read content that is repeated on each page and cannot skip past it, their experience on the page can be very frustrating. A common method used to bypass repetitive content is internal (same page) links. **Note:** • Like other controls, the skip-navigation link must be keyboard navigable and receive visible focus.
Related Standards	508 1194.22(o): Method to Skip Repetitive Links WCAG2: 2.4.1 Bypass Blocks
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find repeated blocks of content and/or repetitive navigation links (menus, for example).

Test Instruction 2: Inspecting/Using Components	○ Use the WAT (Doc Info - Skip Link) to reveal instances of skip links and their targets. ○ The skip target must be located after the repetitive content. ○ If no skip links are marked, another skip method may have been used. TAB toward the repetitive blocks of content/navigation links to find a skip function. The skip function may only reveal when it receives keyboard focus. b. Test the functionality of the Skip method: ○ Activate the function by keyboard.TAB to the link, activate it with the ENTER key, TAB again. ○ Determine if the focus was moved past the repeated content. The visual focus should have shifted to an interactive element after the repetitive content. ○ If there is no interactive element after the repetitive content, the focus would shift to the browser. To determine the actual location of the skip target, adjust the browser window height (smaller) to force a visual shift of content within the browser window when the skip function is activated. **Notes:** • If Skip links are there but they are not working properly, this is a failure. • If there is a need for multiple skip links on a page, each skip link must describe its purpose to comply with the links requirement (#16). For example, a page with repetitive links should have a skip link to jump past these links. If there is also repetitive content, this should have a separate skip link. • Repeated content that is contained in its own separate frame is not included in this test.
Test Instruction 3a: Section 508 Failure Conditions	• There is no method to skip past repeated blocks of content or links. ○ Fails 1194.22(o): Method to Skip Repetitive Links.
Test Instruction 3b: WCAG2 Failure Conditions	• There is no method to skip past repeated blocks of content or links. ○ Fails 2.4.1 Bypass Blocks
Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a ○ Fails Baseline Requirement #5 • There are repeated blocks of content or links and there are skip-links. ○ Passes Baseline Requirement #5 • There are no repeated blocks of content or links. ○ Not applicable (Baseline Requirement #5)
Advisory: Tips for streamlined test processes	• If Skip-navigation links are there but the keyboard cannot be used to shift the focus, this is a failure of the Keyboard test (#1). • If Skip-navigation links are there but they are not visible when focused, this is a failure of the Focus (visible) test (#2).

6. Multi-state components

Requirement	6. Components that can change their state must reveal their current state and function to Assistive Technology.
Rationale	Certain components in an interface can change their state. States include such things as closed/open, ascending-order/descending-order, collapsed/expanded. Dynamic values can also be shown on components (e.g., "34 characters remaining", "Alert Priority 5"). The current state and function of interface components is usually visually apparent. However, these characteristics of the component must be provided and discoverable by assistive technology for users without vision or with low vision (including without color perception).
Related Standards	508 1194.21(d): Name, Role, State. 508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG 1.3.1. Info and Relationships WCAG2: 3.2.1 On Focus WCAG2: 3.2.2 On Input WCAG2: 4.1.2 Name, Role, Value
Tools Necessary	WAT, ARIA Markup Favelet, Inspect/Java Ferret
Test Instruction 1: Finding Applicable Components	a. Find components that indicate status or can change their state. Examples include images, tree navigation, data table sort functions. **Note:** • It may be necessary to use the mouse to determine whether state changes occur on hover, or on click.

Test Instruction 2: Inspecting/Using Components	a. Inspect components to find associated information using the following methods (either in combination or singularly): ○ Read Screen Text on a component ○ **[Web only]** Use WAT to reveal Titles (Doc Info - Show Titles) ○ **[Web only]** Use WAT to reveal ALT attributes (Images – Show Images) ○ **[Web only]** Use ARIA Markup Favelet to reveal ARIA attributes. The ARIA attribute may contain the text description or reference text on the page. ○ *[SW only]* Use Inspect/Java Ferret to examine the Name, Role and State of components b. Determine whether the current State and function are correct. Consider the following test criteria: ○ Sometimes information might be combined (e.g., Name and State are provided together, such as "tree view expanded", or the function and State are the same, such as "34 characters remaining") ○ The component must unambiguously give its current state, rather than what its state would be after a change is activated (e.g., "Submenu, closed" is unambiguous, whereas "Close Submenu" sounds like an instruction). It may be necessary to change the state of components to check that this is working properly. ○ Complete status information is required. If a data table offers multiple sort options (for example, sortable by date, last name, and city), the data table's current status must include which column is the primary sort option and how that column is sorted. It is not required that a single component provide the complete status for a component (e.g., an asterisk can indicate the column that is the primary sort and a down arrow can indicate that it is sorted alphabetically A to Z). • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.
Test Instruction 3a: Section 508 Failure Conditions	• *[SW only]* A multi-state component does not reveal its current information (Name, Role, and/or State). ○ Fails 1194.21(d): Name, Role, State. • **[Web only]** A multi-state component does not reveal its current information (ALT, TITLE, ARIA attributes). ○ Fails 1194.31(a): Use without vision. ○ Fails 1194.31(b): Use with low vision.
Test Instruction 3b: WCAG2 Failure Conditions	• A multi-state component does not reveal its current information (Name, Role, and/or State). ○ Fails 1.3.1 Info and Relationships ○ Fails 3.2.1 On Focus ○ Fails 3.2.2 On Input ○ Fails 4.1.2 Name, Role, Value

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #6
	• Multi-state components reveal their current information (Name, Role, and/or State).
	  ○ Passes Baseline Requirement #6
	• There are no components that can change their state.
	  ○ Not applicable (Baseline Requirement #6)

Advisory: Tips for streamlined test processes	• Addressing content that updates somewhere on the web page or software screen other than the current area of focus (i.e., non-user-initiated state change) is not addressed in the baseline tests. It may be worth examining/addressing. • **[Web only]** For images, the preferred method is ALT text (unless input fields or links), but TITLE is allowed in this baseline test. • **[Web only]** A user guide for ARIA states may be helpful. In some cases, the state may flip when using the ARIA Markup Favelet. • If ARIA attributes other than 'labeled-by', 'described-by', and ‘required=true’ are used, further testing with a screen reader may be needed to provide conclusive results.

7. Images

Requirement	7. All images must have associated text describing the purpose and/or function of the image. Decorative images do not require a description.
Rationale	Screen reader software cannot interpret images. The software will, however, read text that has been associated with images. The interpretation (meaning) of an image must therefore be conveyed textually in the interface programming. The meaning of visual information is inherently contextual. For example, a picture of a person running on a page about athletics is contextually different to the same picture of a person running on a page about data connection speeds. Therefore, instead of just describing a picture ("person running") a description is needed in context ("Come join the athletics team" versus "With our network speeds, you'll be ahead of the race"). Images of text are sometimes used instead of screen text to achieve an artistic effect. When text is rendered as an image, the Alternate text should be the same words verbatim. If font-based graphics are used to provide information, equivalent information must be provided in an accessible format. Images that are used a number of times throughout an application (e.g., icons on navigation controls) must have a consistent meaning and text description throughout the application. For example, if an icon of a blank piece of paper means "new document" on most screens, the same icon cannot be used elsewhere to mean "reformat document". Consistency aids users with cognitive disabilities. Some images and animations are decorative and convey no information. Decorative components do not need a description, but they do need a tag to affirm to the user that there is no content in the image. **Note:** • **[Web only]** The description is most often provided as Alternate text ("ALT text") attribute on an image. It is also acceptable to use a TITLE attribute for the description. If both ALT and TITLE are provided for an image, the review of the ALT should take precedence.
Related Standards	508 1194.21(d): Name, Role, State 508 1194.21(e): Bitmap images 508 1194.22(a): Equivalent text descriptions WCAG2: 1.1.1 Non-text Content WCAG2: 3.2.4 Consistent Identification
Tools Necessary	WAT, Inspect/Java Ferret

Test Instruction 1: Finding Applicable Components	a. Reveal where images have been used: ○ Look for images that are rendered by using font-based graphics (e.g. up-arrows to indicate sort order, etc.). ○ **[Web only]** Use the WAT (Select Images – Show Images) to on the page. Some components may not look like images, but they will show up in the WAT output. b. *[SW only]* Look for components that appear to be images. Determine whether images are decorative. An image may be considered decorative if it is purely artistic, or if it is redundant with the text information next to it (e.g., the caption includes the purpose and/or the function of the image).
Test Instruction 2: Inspecting/Using Components	a. Reveal descriptive text on images: ○ **[Web only]** Use the WAT (Select Images – Show Images) to reveal the ALT content and (Select Doc Info – Show Titles) to reveal descriptive title attributes. ○ *[SW only]* Use Inspect/Java Ferret Name property to check for an equivalent text description. b. Examine the descriptive text to determine whether the purpose and/or function of the image has been conveyed for all non-decorative images. It may be necessary to check the surrounding text and other content to determine whether the descriptive text makes sense in context. c. Examine the descriptive text on text rendered as an image to check whether the texts match verbatim. d. Examine the descriptive text on all decorative images: ○ **[Web only]** There must be a null ALT on each decorative image (ALT="") ○ *[SW only]* The Name should equal 'None' on each decorative image found with Inspect (decorative images not found with Inspect may be safely ignored e. If there are any CAPTCHA images, the descriptive text should describe the purpose of the image, not the text of the CAPTCHA. f. Examine instances where the same image is used multiple times. Check that the meaning of the image (conveyed through visual appearance as well as descriptive text) is consistent throughout the application. **Note:** • If data charts contain a great deal of detail, the image may be supported by a data table underneath the chart, or linked to underneath the chart. The ALT on the chart can then be the name of the chart, the pertinent trends displayed in the chart (see also the surrounding text for context), or a combination of both. • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• CAPTCHA descriptive text does not contain the purpose of the CAPTCHA.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• Inconsistent meaning (visual appearance and/or descriptive text) on images used multiple times.
	  ○ Fails 1194.21(e): Bitmap images.
	• The descriptive text on text rendered as an image does not match verbatim
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** The purpose and/or function of a non-decorative image is not properly conveyed in descriptive text.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** Missing ALT text or TITLE on a non-decorative image.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** Missing ALT="" on a decorative image.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** ALT containing a description on a decorative image.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• *[SW only]* The purpose and/or function of a non-decorative image is not properly conveyed in the Name property
	  ○ Fails 1194.21(d): Name, Role, State.
	• *[SW only]* Decorative images found by Inspect have do not have a 'None' Name property.
	  ○ Fails 1194.21(d): Name, Role, State.

Test Instruction 3b: WCAG2 Failure Conditions	• CAPTCHA descriptive text does not contain the purpose of the CAPTCHA.
	  ○ Fails 1.1.1 Non-text Content.
	• Inconsistent descriptive text on images used multiple times.
	  ○ Fails 3.2.4 Consistent Identification
	• The descriptive text on text rendered as an image does not match verbatim
	  ○ Fails 1.1.1 Non-text Content.
	• The purpose and/or function of a non-decorative image is not properly conveyed in descriptive text.
	  ○ Fails 1.1.1 Non-text Content.
	• Missing ALT text or TITLE on a non-decorative image.
	  ○ Fails 1.1.1 Non-text Content.
	• Missing ALT="" on a decorative image.
	  ○ Fails 1.1.1 Non-text Content.
	• ALT containing a description on a decorative image.
	  ○ Fails 1.1.1 Non-text Content.

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #7
	• Images have an ALT-Text or Title attribute AND the meaning, and/or purpose of the image is sufficiently described, AND the meaning of images used multiple times is consistent.
	  ○ Passes Baseline Requirement #7
	• There are no images.
	  ○ Not applicable (Baseline Requirement #7)

Advisory: Tips for streamlined test processes	• Images are prime candidates for providing extensive guidance and examples on how to provide alternate/descriptive text. It is a subjective task, requiring consideration of a number of factors. There are numerous common mistakes to watch out for. • If the WAT (Select Images – Show Images) command does not mark an image on a web page, it may be that the image is generated via CSS, which would be covered under Baseline Test #22. • Advice on handling CAPTCHA images can cover the descriptive text for the CAPTCHA, and add that the function on the CAPTCHA should still be accessible (by following other Baseline tests as required). • **[Web only]** The description is more often provided as Alternate text, and is preferable to using the TITLE attribute (although the use of either or both is allowed).

8. Color (meaning)

Requirement	8. Color must not be the only means of conveying information, indicating an action, prompting a response, or indicating status. Information conveyed through color must also be provided in text displayed on the screen or by visual differentiation.
Rationale	Color dependence is defined as using color as the sole means to convey information. For example, a single indicator that is green for 'on', orange for 'standby', and red for 'off' is color dependent. When color is the only means to convey information, people who are color blind, and people who cannot see, do not have access to the same information that others have. The status or function that is being conveyed by color also needs to be available in a textual format that can be viewed by all, and can be read by screen reader software. This requirement does not mean that color cannot be used; it means that color cannot be the only means of conveying the information.
Related Standards	508 1194.21(i): No color dependence to convey information 508 1194.22(c): No color dependence to convey information WCAG2: 1.1.1 Non-text Content WCAG2: 1.4.1 Use of Color
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find where color conveys meaning, indicates an action, or prompts a response.
Test Instruction 2: Inspecting/Using Components	a. Where color is used to convey meaning, determine if meaning is present via: ○ Screen text, displayed when the meaningful color is displayed, describing the color (e.g. the word "ALERT" for a red indicator, or an asterisk for a required field), or ○ Visual differentiation (e.g., shape, position or size). b. **[Web only]** Use WAT (Colour - Greyscale) where content is suspect in terms of color dependency, to determine whether the meaning is clear when color is not used.
Test Instruction 3a: Section 508 Failure Conditions	• *[SW only]* An instance of color being the sole means of conveying meaning. ○ Fails 1194.21(i): No color dependence to convey information. • **[Web only]** An instance of color being the sole means of conveying meaning. ○ Fails 1194.22(c): No color dependence to convey information.
Test Instruction 3b: WCAG2 Failure Conditions	• An instance of color being the sole means of conveying meaning. ○ Fails 1.1.1 Non-text Content. ○ Fails 1.4.1 Use of Color.

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #8
	• Color is used to convey meaning AND the same information is provided via screen text or visual differentiation.
	  ○ Passes Baseline Requirement #8
	• Color is not used to convey meaning.
	  ○ Not applicable (Baseline Requirement #8)

Advisory: Tips for streamlined test processes	• When color is used to communicate data sets (e.g., Geographic Information System application, or pie chart), additional guidance may be necessary on testing for equivalent facilitation. Related tests might include multi-state components, images, links, and Alternate pages. Data tables related to the charts may also be suggested means of augmenting the standard interface. • For Web testing, the WAT tool may sometimes return a page with missing information, due to the program's limitations. In such cases, and with Software testing, it may be necessary to conduct a manual inspection such as printing a screen capture on a black and white printer. • WAF's Greyscale test works in more cases than WAT's, although there may be some sites where both work only partially or not at all.

9. Color (contrast)

Requirement	9. There must be contrasting colors/shades at a minimum ratio of 4.5:1 for discerning between background and foreground content.
Rationale	The visual difference between the background behind text, and the text itself, may be perceivable by a given designer. However, beyond color choice which is under control of the designer, many factors beyond the designer's control affect peoples' ability to discern between colors/shades, including age (contrast sensitivity reduces with age),screen brightness, ambient light, color blindness and some types of low vision. The use of color/shade choices that do not contrast well with each other may be deliberate (i.e., artistic preference), or they may be the result of programmatic features (e.g., a button's text is black on white, but the text turns yellow in a certain mode, and the background remains white). In general, the higher the level of contrast used, the more people will be able to see and use the content.
Related Standards	508 1194.31(b): Use with low vision WCAG2: 1.4.3 Contrast (Minimum)
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Visually examine all appearances of meaningful text and images of meaningful text displayed on the page for areas that may have low background to foreground contrast.
Test Instruction 2: Inspecting/Using Components	a. Use WAT (Colour – Contrast Analyser (application)) and use the colour picker tool to select foreground and background colors from the screen. Select the Luminosity Algorithm. WAT will display a luminosity contrast ratio, which must be at least 4.5:1. **Notes:** • The Contrast Analyser test is a rudimentary and does not address all users with reduced contrast sensitivity. In cases where certain color/shade combinations are suspect, it may be necessary to utilize additional tools. • *[SW only]* Text contained in Logos is exempt from this requirement.
Test Instruction 3a: Section 508 Failure Conditions	• An instance of colors/shades for discerning between background and foreground content having contrast ratios of less than 4.5:1 ○ Fails 1194.31(b): Use with low vision.
Test Instruction 3b: WCAG2 Failure Conditions	• An instance of colors/shades for discerning between background and foreground content having contrast ratios of less than 4.5:1 ○ Fails 1.4.3 Contrast (Minimum).

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #9
	• Colors/shades for discerning between background and foreground content have contrast ratios of 4.5:1 or better.
	  ○ Passes Baseline Requirement #9

Advisory: Tips for streamlined test processes	• Text contained in Logos rendered as images is exempt from this requirement, but there must be an ALT-text for the logo. • The thresholds in the Colour Contrast Analyser are based on the WCAG 2 minimum contrast ratio of 4.5:1. WCAG 2.0 recommends a lower threshold of 3:1 for 'large scale' text (18 point or 14 point bold). Although the Colour Contrast Analyser has a pass/fail indicator for large text, it does not determine the size of the text being tested. It is acceptable to allow a 3:1 ratio for larger text so long as the test includes a reliable mechanism for determining the font's point size. • The WAT contrast analyser is not specific to web content only. The tool can be used for software inspection in any window. • If using a dual-monitor setup, it is necessary to have both the tool and the window under test displayed on the primary monitor. • Instructions for testing of text contrast changes due to mouse hover and status can be incorporated into streamlined tests. • Incidental text is exempt from this requirement. Text or images of text that are part of an inactive user interface component, that are pure decoration, that are not visible to anyone, or that are part of a picture that contains significant other visual content, have no contrast requirement. An example of incidental text is disabled form field labels.

10. Flashing (reserved)

Requirement

10. Sections(s) of the screen should not flash at or above 3Hz.

Note:
Agencies must include an evaluation of flashing/blinking content in their test processes. However, as of the publication of the current version of baseline tests, there is no agreed-upon testing method.

For more information and advisory notes, see the attachment at the end of this document.

11. Forms (associated instructions)

Requirement	11. Labels, instructions, directions and cues necessary to complete a form must be programmatically associated with their respective input control.
Rationale	In order to correctly and accurately complete a form, it is necessary to follow instructions, directions and cues, as well as enter information in the correct places. A given form component may be the subject of instructions that are not positioned next to the component (e.g., at the top of a form, the instruction is "If you are the home owner, complete parts a, b, and f"). In such cases, form designers will use visual layout and flow to direct the user. However, users without vision, or with low vision, may not have access to the visual cues, and hence will be unable to easily find the related instructions for the current form component. For this reason, it is necessary to programmatically associate all relevant instructions, directions and cues with their respective components/controls. **Note:** • Read-only (e.g. pre-filled) form fields are considered interactive, in that they need to receive keyboard focus and must be labeled.
Related Standards	508 1194.21(f): Input text 508 1194.21(l): Forms 508 1194.22(n): Labels for forms WCAG2: 1.3.1 Info and relationships WCAG2: 3.3.2 Labels or instructions
Tools Necessary	WAT, ARIA Markup Favelet, Inspect 32/Java Ferret
Test Instruction 1: Finding Applicable Components	a. Find all form input components. Examples include buttons, text fields, radio buttons, checkboxes, multi-select lists (combo boxes). b. Find all instructions and cues (textual and graphical) that are related to form components/controls, including groupings, order of completion, special conditions or qualifiers, etc.
Test Instruction 2: Inspecting/Using Components	a. **[Web only]** Inspect the form to see if all instructions and cues are present within labels. Use WAT (Structure – Fieldset/Labels, Doc Info - Show Titles): ○ TITLE attributes can be used to duplicate the visual label, which can be rendered as screen text or as an image (duplicating the label in the TITLE attribute can aid screen reader users, but is not required). ○ The TITLE attribute cannot be the sole means of providing information. b. **[Web only]** Check that LABEL and ID are valid HTML: ○ Check that there is only one 'LABEL FOR' and only one 'ID' assigned to each instruction / input component. ○ Every ID on the page must be unique. ○ The LABEL FOR and ID for the pair must be identical (case sensitive). ○ Determine whether the page is coded in HTML 4.01 or HTML5. Press F12 or view source to inspect the DOCTYPE. ▪ <!DOCTYPE HTML> means the page is coded in HTML5.

	▪ <!DOCTYPE HTML PUBLIC … > means the page is coded in HTML 4.01. ○ In HTML 4.01: ▪ IDs must start with a letter (starting with numbers is prohibited). ▪ After the first letter any number of letters (a to z, A to Z), digits (0 to 9), hyphens (-), underscores (_), colons (:) and periods (.) are allowed. ○ In HTML5: ▪ IDs can have any value as long as it is unique, not the empty string, and does not contain spaces. c. **[Web only]** If all instructions and cues are present and related to the components, this completes this test. If this is not the case, continue to step d. d. **[Web only]** Use the ARIA Markup Favelet. ○ Determine whether ARIA form attributes exist on the page. ○ **[Web only]** Check the ARIA form attributes with Inspect. Check for Name, Legacy Role, Legacy State, and ARIA properties. The Name property must be complete and include all form labels identified in the ARIA properties. (If ARIA-REQUIRED is used to indicate required fields, it will be listed in the "AriaProperties" property.) *[SW only]* Inspect the form to see if all instructions and cues are present within labels. Use Inspect/Java Ferret to examine the Name, Role, State, and Value information for each component. ○ Tab through all interactive interface components. ○ The Name of each component should match the visual label. ○ The Role of each component must accurately reflect its function (e.g., button, slider, checkbox, edit field). ○ The Value property must be correct for text input form fields. ○ Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content. ○ Forms with ARIA labels should be tested in Chrome or Firefox to determine the accessibility of the coded forms. ○ The State of components with focus must be "focused, focusable". The State of checkboxes and radio buttons must include whether the option is currently selected/checked or unselected/unchecked. **Notes:** • There are many ways to indicate that a field is required. Usually this is visually indicated by a star (*). This information needs to be directly associated with the input component via one of the above methods (adding a title attribute, including the '*' in the label, or through ARIA 'required=true'). • All error handling and error suggestions should be considered as instructions and cues necessary to complete the form. This is regardless of whether the errors are detected before or after a submission/page reload or when errors are handled as the user enters information (error handling can be addressed in many ways). If error messages and alerts receive focus, then error instructions and cues do not need to be programmatically associated with a form field.

Test Instruction 3a: Section 508 Failure Conditions	• *[SW only]* The Name of a component does not match its visual label
	  ○ Fails 1194.21(l): Forms.
	• *[SW only]* The Role of a component does not accurately reflect its function.
	  ○ Fails 1194.21(l): Forms.
	• *[SW only]* The State of components with focus is not "focused, focusable".
	  ○ Fails 1194.21(l): Forms.
	• *[SW only]* The State of checkboxes and radio buttons does not show that an item is selected correctly.
	  ○ Fails 1194.21(l): Forms.
	• *[SW only]* Instructions and cues (including whether a field is considered 'required') are not related to their components.
	  ○ Fails 1194.21(l): Forms.
	• *[SW only]* The Value property is incorrect for text input fields.
	  ○ Fails 1194.21(f): Input text
	• **[Web only]** LABEL FOR and ID are used, but not as valid HTML
	  ○ Fails 1194.22(n): Labels for forms.
	• **[Web only]** For a form field, the TITLE attribute is the sole means of providing information (i.e., there is no additional screen text or visual context).
	  ○ Fails 1194.22(n): Labels for forms.
	• **[Web only]** Instructions and cues are not related (through LABEL, TITLE, or ARIA attribute) to their respective input components.
	  ○ Fails 1194.22(n): Labels for forms.

Test Instruction 3b: WCAG2 Failure Conditions	• The Name of a component does not match it's visual label
	  ○ Fails 1.3.1 Info and relationships.
	  ○ Fails 3.3.2 Labels or instructions.
	• The Role of a component does not accurately reflect its function.
	  ○ Fails 1.3.1 Info and relationships
	  ○ Fails 3.3.2 Labels or instructions.
	• The State of checkboxes and radio buttons does not show that an item is selected correctly.
	  ○ Fails 1.3.1 Info and relationships
	• Instructions and cues are not related to their components.
	  ○ Fails 1.3.1 Info and relationships.
	  ○ Fails 3.3.2 Labels or instructions.
	• The Value property is incorrect for text input fields.
	  ○ Fails 1.3.1 Info and relationships
	• Label for and ID are used, but not as valid HTML
	  ○ Fails 1.3.1 Info and relationships.
	  ○ Fails 3.3.2 Labels or instructions.
	• For a form field, the TITLE attribute is the sole means of providing information (i.e., there is no additional screen text or visual context).
	  ○ Fails 3.3.2 Labels or instructions.
	• Instructions and cues are not related (through label, title, or ARIA attribute) to their respective input components.
	  ○ Fails 1.3.1 Info and relationships
	  ○ Fails 3.3.2 Labels or instructions.

Test Instruction 3c: Baseline	• Any failure in 3a ○ Fails Baseline Requirement #11

Requirement Test Results	• Labels, instructions, directions and cues necessary to complete a form are correctly associated with their respective input components programmatically.
	  ○ Passes Baseline Requirement #11
	• There are no form input components.
	  ○ Not applicable (Baseline Requirement #11)

Advisory: Tips for streamlined test processes	• Forms still have to be covered by all other tests that are applicable (e.g., Keyboard, focus (visible), focus (order)). • Instructions for interpreting and assessing the 'Fieldset' tags that are revealed by WAT can be incorporated into streamlined tests. • If ARIA attributes other than 'labeled-by', 'described-by', and ‘required=true’ are used, further testing with a screen reader may be needed to provide conclusive results. • It should be noted for testers that when using Inspect32, the properties are listed in MSAA mode as "Role", "Name", etc.; but in UI Automation mode they are listed a "LegacyIAccessible.Role", "LegacyIAccessible.Name", etc. • In testing form fields with JavaFerret, it may be necessary to type into the field, then TAB out of the field, then SHIFT+TAB back into the field in order to reveal the Value property. • Examples of code and screenshots with testing tool results are highly recommended to help testers.

12. Page Titles

Requirement	12. Programmatically identify Page Titles.
Rationale	Page titles appear in the title bar of the browser or software window (and in the tabs where multiple tabs in a single window are used). If there is no programmatically defined page title, visually capable users can assimilate the content quickly to know where they are. However, non-visual users will have to navigate through the content to know what page they are on, which can take an undue amount of time. Screen reader technologies will announce the programmatically defined page title when the user is browsing between tabs and between windows.
Related Standards	508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 2.4.2 Page titled
Tools Necessary	None
Test Instruction 1: Finding Applicable Components	a. Examine the title bar and/or the tab of the current page or software window.
Test Instruction 2: Inspecting/Using Components	a. Check that the page title is a meaningful representation / indication of the content. The title should be in plain language (rather than code). **Note:** • *[SW only]* Some software may not use the title bar. It may be necessary to use ALT-TAB to cycle through open applications, or look in the Windows Taskbar.

Test Instruction 3a: Section 508 Failure Conditions	• No page title in plain language.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• No page title in plain language. ○ Fails 2.4.2 Page titled

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #12
	• There is a page title in plain language.
	  ○ Passes Baseline Requirement #12

Advisory: Tips for streamlined test processes	• Standard software convention is to use OS platform components which include an application title bar. Some applications may not use the title bar, but should still be required to present their names to AT when switching between applications.

13. Data Tables (headers)

Requirement	13. Column and row header cells of data tables must be programmatically identified.
Rationale	For users with vision, the process of determining what headers go with a data cell is usually straightforward, especially when formatting such as bold letters and shading is applied to the headers. For users of screen reading software, however, things like 'bold' and 'shaded' have no useful meaning, so using styles and formatting to identify headers does not work. Instead, row and column headers must have programmatic markup to enable them to be identified by the screen reading software. On all data tables column and row headers must be identified. **Notes:** • Data tables are those tables where the information in a cell requires a row or column header to adequately describe the cell's contents. If a table is used for placement of components on the page for visual aesthetics, then it is a layout table. This test applies to data tables only. • Complex data tables are defined as those that have two or more levels of headers, and/or include split or merged cells.
Related Standards	508 1194.21(d): Name, Role, State 508 1194.22(g): Identify row and column headers WCAG2: 1.3.1 Info and Relationships
Tools Necessary	WAT, Inspect
Test Instruction 1: Finding Applicable Components	a. Find data tables: ○ **[Web only]** Use WAT (Tables - Table Borders) to find where table markup has been used. Identify which tables are data tables (to test) and which are layout tables (to ignore). ○ *[SW only]* Visually inspect the content to find data tables. **Note:** • **[Web only]** If color is used extensively on the page, the table borders may not be easily distinguishable using the above WAT test. If this is the case, use WAT (Tables - Show Data Tables).

Test Instruction 2: Inspecting/Using Components	a. Identify all of the headers for each data cell (some cell headers may not be in the same row and/or column as the data cell). b. Reveal the markup assigned to row and column headers: ○ **[Web only]** Use WAT (Tables – Show Data Tables). Each row and column header must have either a SCOPE="row | col", SCOPE="rowgroup | colgroup", or an ID="x". If ID is used, data cells must refer to the associated header cell's ID in order for the header to pass this test. ○ *[SW only]* Use Inspect to examine each header cell. The Name must match the screen text. The Role must be accurate (table, column header, or row header). Notes: • **[Web only]** Only data tables with one or two header levels may employ the SCOPE="row | col | rowgroup | colgroup" attributes if all headers are in the same row and/or column or group as the data cell. Data tables with more than two header levels, or with headers that are not in the same row and/or column as the data cell, must use HEADER/ID. (ID can be used on a table with any number of header levels, and complex tables.) • An image of a data table, with its contents described in Alternate text, will fail this test. Data tables must be marked up programmatically. • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content. • **[Web only]** For tables that use TD SCOPE, determine whether the page is coded in HTML 4.01 or HTML5. Press F12 or view source to inspect the DOCTYPE. ○ <!DOCTYPE HTML> means the page is coded in HTML5. ○ <!DOCTYPE HTML PUBLIC … > means the page is coded in HTML 4.01. ○ In HTML5,TD SCOPE is not supported (only TH SCOPE is supported).
Test Instruction 3a: Section 508 Failure Conditions	• **[Web only]** A data table does not have its row and/or column header cells correctly marked programmatically. ○ Fails 1194.22(g): Identify row and column headers. • *[SW only]* A data table does not have its row and/or column header cells correctly marked programmatically. ○ Fails 1194.21(d): Name, Role, State.
Test Instruction 3b: WCAG2 Failure Conditions	• A data table does not have its row and/or column header cells correctly marked programmatically. ○ Fails 1.3.1 Info and Relationships.
Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a ○ Fails Baseline Requirement #13 • Data tables have their row and/or column header cells correctly marked programmatically. ○ Passes Baseline Requirement #13 • There are no data tables. ○ Not applicable (Baseline Requirement #13)

Advisory: Tips for streamlined test processes	• Worked examples would be a great help for testers. • Testing should include data table cell mappings (Baseline Test #14). • In HTML 4.01, TH SCOPE is preferred over TD SCOPE for headers. However, TD is acceptable at this time. • A calendar / date picker that has headers (Sun, Mon, Tues etc.) may need to be treated as a data table. The visual information must be provided programmatically for each element of the calendar (year, day, month, blackout days, etc.). • Although not prohibited by this test, data table structural elements such as 'TH' should not appear in Layout tables, as it is an example of bad coding. Adding ARIA 'Role=presentation' can solve this coding problem. See also Note F46 under 'Understanding WCAG 1.3.1' on the WCAG website

14. Data Tables (cell-header association)

Requirement	14. Data cells on complex tables must include markup to associate the data cell with the correct header.
Rationale	Complex data tables are defined as those that have two or more levels of headers, and/or include split or merged cells. On complex tables, markup is needed to define which headers are associated with data cells, so that screen reader users can determine where they are for any given cell or set of cells.
Related Standards	508 1194.21(d): Name, Role, State 508 1194.22(h): Associate Data with Headers WCAG2: 1.3.1 Info and Relationships
Tools Necessary	WAT, Inspect
Test Instruction 1: Finding Applicable Components	a. Find data tables: ○ **[Web only]** Use WAT (Tables - Table Borders) to find where table markup has been used. Identify which tables are data tables (to test) and which are layout tables (to ignore). ○ *[SW only]* Visually inspect the content to find data tables. **Notes:** • **[Web only]** If color is used extensively on the page, the table borders may not be easily distinguishable using the above WAT test. If this is the case, use WAT (Tables - Show Data Tables). • If a data cell's header is not in its row or column, treat it as a complex table.

Test Instruction 2: Inspecting/Using Components	a. Identify all of the headers for each data cell (some cell headers may not be in the same row and/or column as the data cell). b. Reveal the markup assigned to data cells: ○ **[Web only]** Use WAT (Tables – Show Data Tables). ○ *[SW only]* Use Inspect to examine each data cell. The Name must match the screen text and include column and row header information. The Role must be accurate (cell c. **[Web only]** Where a data table has a maximum of 2 levels of headers, SCOPE (row/col/rowgroup/colgroup) is acceptable if all headers are in the same row and/or column or group as the data cell. d. **[Web only]** Check any data table that should identify its column or row headers with ID or SCOPE: ○ Either: ▪ Each data cell must have a TD HEADERS="x y", where "x y" refers to the correctly associated headers (TH ID="x" on one header and ID="y" on the second header) ○ Or: ▪ Each data cell is correctly associated to all its relevant headers by use of SCOPE="row | col | rowgroup | colgroup". ○ Where cell headers are not in the same row and/or column as the data cell, check that each ID / HEADER association is correct. e. **[Web only]** In HTML5, TD SCOPE is not supported (only TH SCOPE is supported). ○ If TD SCOPE is used, determine whether the page is coded in HTML 4.01 or HTML5. Press F12 or view source to inspect the DOCTYPE. ▪ <!DOCTYPE HTML> means the page is coded in HTML5. ▪ <!DOCTYPE HTML PUBLIC … > means the page is coded in HTML 4.01. ○ In HTML 4.01: ▪ IDs must start with a letter (starting with numbers is prohibited). ▪ After the first letter any number of letters (a to z, A to Z), digits (0 to 9), hyphens (-), underscores (_), colons (:) and periods (.) are allowed. ○ In HTML5: ▪ IDs can have any value as long as it is unique, not the empty string, and does not contain spaces. f. Where data cells are associated with multiple headers (e.g., Headers are 'Sales... January... Pending', and data is '575'), each header association must be explicitly made in each data cell. **<u>Note:</u>** • An image of a data table, with its contents described in Alternate text, will fail this test. Data tables must be marked up programmatically. • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• **[Web only]** A complex data table has data cells that do not have associations with header cells correctly marked programmatically.
	  ○ Fails 1194.22(h): Associate Data with Headers.
	• *[SW only]* A complex data table has data cells that do not have associations with header cells correctly marked programmatically.
	  ○ Fails 1194.21(d): Name, Role, State.

Test Instruction 3b: WCAG2 Failure Conditions	• A complex data table has data cells that do not have associations with header cells correctly marked programmatically. ○ Fails 1.3.1 Info and Relationships.

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #14
	• Data cells of complex tables have their associations with header cells correctly marked programmatically.
	  ○ Passes Baseline Requirement #14
	• There are no data tables.
	  ○ Not applicable (Baseline Requirement #14)

Advisory: Tips for streamlined test processes	• Worked examples would be a great help for testers. • Testing should include data table headers.

15. Headings

Requirement	15. **[Web only]** Headings must be programmatically identified and must match the visual outline level.
Rationale	Headings are used to visually and semantically break up content to make it easier to read, easier to find and understand relevant information, and so on. Headings can be visually marked using text formatting such as bold, underline, or combinations (e.g., bold, underlined, and large font means a major heading). Screen reader technologies cannot automatically infer meaning from formatting changes. A given piece of text may be in italics because it is emphasizing a point, or because it is a heading. Because there is no way to infer meaning, headings can use visual formatting, but they must also be programmatically identified for identification with AT. **Notes:** • The requirement should not be construed to require headings in place of headers in data tables. • This requirement does not mean that headings be added; it means that where headings are identifiable through visual formatting, they must be programmatically identified. • Any visual representations of heading level (e.g., major section, section, subsection) must be matched by the programmatic heading level (i.e., major section = level 1, section = level 2, sub-section = level 3). Matching the programmatic level with the visual level is essential for proper comprehension of the content for non-visual users.
Related Standards	508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 1.3.1 Info and Relationships
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Visually identify where headings are used on the page, through formatting or use of white space, boxes or other visual separators.
Test Instruction 2: Inspecting/Using Components	a. Use the WAT (Structure - Headings, Structure - Heading Structure) to reveal headings and the hierarchy used. b. Check that any visual headings are also marked as headings, <H1>, <H2> etc. c. Check that the Heading levels match the visual structure.

Test Instruction 3a: Section 508 Failure Conditions	• Visually apparent headings are not programmatically identified.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.
	• Programmatically identified heading levels do not match the visual outline level.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• Visually apparent headings are not programmatically identified.
	  ○ Fails 1.3.1 Info and Relationships
	• Programmatically identified heading levels do not match the visual outline level.
	  ○ Fails 1.3.1 Info and Relationships

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #15
	• Visually apparent headings are programmatically identified AND heading levels match the visual outline level.
	  ○ Passes Baseline Requirement #15
	• There are no visually apparent headings.
	  ○ Not applicable (Baseline Requirement #15)

Advisory: Tips for streamlined test processes	• If a page appears to have logical separable sections, but there are no headings, it might be worth pointing out to the authors that identifying such sections through headings might be useful for all users. • HTML section headings are used to provide structure on a web page, facilitating faster comprehension. However, some designers may use heading tags for non-heading purposes, such as text styling to call visual attention to content. Such uses deviate from the primary purpose of headings, which is to provide information on how the content on the page is structured. It might be worth pointing out to the designers that using heading tags for non-headings can cause confusion for non-visual users. • Examples would be helpful to illustrate headings that do and do not match their visual structure.

16. Links and User controls

Requirement	16. Links and/or user controls must have meaningful names that describe the unique destination, function, and/or purpose of the control for assistive technology.
Rationale	To aid navigation with screen reading AT software, users can call up a list of links on a web page or software screen. Users can read through content and decide which of the links in the content they wish to follow (i.e., they do not have to navigate back to the link itself). In order to provide links to end users, there are a number of common methods in practice that render a list of links unhelpful. Say each item for sale has a 'click here' link next to it, and the user calls up the list of links. The list will have multiple 'click here' links that are not distinguishable. Another common problem occurs when the links only contain URLs, and the purpose of each link may not be apparent. It is therefore required to use meaningful and unique names for links and user controls, to aid navigation and use by AT. **Note:** • Links and user controls in image maps are defined by coordinates, and their name and purpose are conveyed through title attribute, or ALT-text attribute.
Related Standards	508 1194.21(d): Name, Role, State 508 1194.22(l): Functional Text for Scripts 508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 2.4.4 Link Purpose (In Context)
Tools Necessary	WAT, Inspect/Java Ferret
Test Instruction 1: Finding Applicable Components	a. Find all links: ○ **[Web only]** Use WAT (Doc Info – List Links) ○ *[SW only]* Scan the content to find links and other user controls. **Notes:** • Some links may contain images. • Some links may be contained in image maps.

Test Instruction 2: Inspecting/Using Components	a. ***Examine each link / user control:*** ○ **[Web only]** Use WAT (Doc Info - List Links) to show on a separate page all link URLs, name and title attribute (if any). ○ *[SW only]* Use Inspect/Java Ferret to examine the Name, Role and State of each link / user control. b. Check that each link / user control on the page has a unique screen text name, or has a non-unique name augmented by a unique title attribute. c. **[Web only]** Use WAT (Doc Info – Javascript/New Window Links) to check for script scripted elements on the page. Check that the elements have a descriptive name in the links list. d. **[Web only]** The method in step a, showing a list of links, helps in checking that link names and titles (if any) are meaningful and unique when spoken in isolation. If there is any doubt that a link is meaningful in the context of surrounding information, other WAT checks can be used to show links and pertinent attributes on the screen, including WAT (Structure - Show Other Elements - "a"), (Doc Info - Show Titles), (Images - Show Images). **Notes:** • **[Web only]** The list of links will include all client-side image map hotspots. The hotspots are links which must be checked in this test. • Step d is to help identify whether a link is meaningful to a user of the application; not whether it is meaningful to the tester. For example, a link named "X17-88.docx" may not make sense to a tester, but in reviewing the rest of the page it may be clear that it would make sense to a typical user of the application. Links that are ambiguous/repetitive ("Click here, Click here, Click here") are covered separately in step b. • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.
Test Instruction 3a: Section 508 Failure Conditions	• A scripted element does not have a descriptive name. ○ Fails 1194.22(l): Functional Text for Scripts. ○ Fails 1194.31(a): Use without vision. ○ Fails 1194.31(b): Use with low vision. • *[SW only]* Software controls do not have a descriptive, unique Name property ○ Fails 1194.21(d): Name, Role, State. • *[SW only]* Software controls have incorrect Role and/or State ○ Fails 1194.21(d): Name, Role State. • The destination, function, and/or purpose of a link / control is not contained in the screen text, name, title attribute, or ALT-text attribute ○ Fails 1194.31(a): Use without vision. ○ Fails 1194.31(b): Use with low vision. • Each link / control is not uniquely identified. ○ Fails 1194.31(a): Use without vision. ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• A scripted element does not have a descriptive name.
	  ○ Fails 2.4.4 Link Purpose (In Context)
	• Controls do not have a descriptive, unique Name property
	  ○ Fails 2.4.4 Link Purpose (In Context)
	• The destination, function, and/or purpose of a link / control is not contained in the screen text, name, title attribute, or ALT-text attribute
	  ○ Fails 2.4.4 Link Purpose (In Context)
	• Each link / control is not uniquely identified.
	  ○ Fails 2.4.4 Link Purpose (In Context)

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #16
	• The destination, function, and/or purpose of the link is contained in the screen text, title attribute, or ALT-text attribute AND each link is uniquely identified.
	  ○ Passes Baseline Requirement #16
	• There are no links or user controls.
	  ○ Not applicable (Baseline Requirement #16)

Advisory: Tips for streamlined test processes	• This test incorporates the link elements within client side image maps, There is a separate test for the existence of server side image maps (#27). • The List of Links provided by the WAT tool may also include a "close window" link which is a WAT function, not a part of the code of the page. This link can be safely ignored in the test. • For Multi-state controls in software, follow the test for multi-state controls.

17. Language

Requirement	17. **[Web only]** A default language must be programmatically identified for each page, and for passages that use a language other than the default. Exceptions: proper names, technical terms, or foreign words that have become part of the vernacular.
Rationale	When a site is in one language but has certain pages that are in a different language, or sections within a page that use a different language, it is necessary to programmatically identify both the default language and the change of language. Screen reader technologies can switch their language pronunciation, but only if there is code to identify the proper language. If language changes are not identified, for a screen reader user, the speech will sound awkward at best, or unintelligible at worst. **Note:** • Interface components that are part of the browser and/or operating system, such as dialog boxes, browser status bar, browser title bar, browser menus, are not included in this test (they are outside of the control of the Web page and its language settings).
Related Standards	508 1194.31(a): Use without vision 508 1194.31(b): Use with low vision WCAG2: 3.1.1 Language of Page WCAG2: 3.1.2 Language of Parts
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Identify the default language of the page, and any passages that differ to the default language.
Test Instruction 2: Inspecting/Using Components	a. Use the WAT (Doc Info - Show Lang Attributes) to reveal the language settings applied to the page and to sections of the page. b. Check that there is an accurate default language attribute set for the page. c. Check that there is an accurate language attribute set for each passage that is different from the page's default language. d. Check that the language attributes match the actual language used. Exceptions: proper names, technical terms, or foreign words that have become part of the vernacular.

Test Instruction 3a: Section 508 Failure Conditions	• The language for the page is not programmatically set.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.
	• A passage (content, image descriptions, form labels etc.) that differs from the default language of the page is not programmatically identified.
	  ○ Fails 1194.31(a): Use without vision.
	  ○ Fails 1194.31(b): Use with low vision.

Test Instruction 3b: WCAG2 Failure Conditions	• The language for the page is not programmatically set.
	  ○ Fails 3.1.2 Language of Parts
	• A passage (content, image descriptions, form labels etc.) that differs from the default language of the page is not programmatically identified.
	  ○ Fails 3.1.2 Language of Parts

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #17
	• The language for the page is programmatically set, AND any passages that differ to the default language of the page are programmatically identified.
	  ○ Passes Baseline Requirement #17

Advisory: Tips for streamlined test processes	• This test is Web only, because in software, language is controlled at the OS platform level.

18. Audio (transcripts)

Requirement	18. Audio-only content must be accompanied by transcripts.
Rationale	Audio-only content includes speeches, and other meaningful audio. Some users will not be able to hear the audio. Therefore, there needs to be a text only version of what is being said, and/or a description of the relevant sounds. **Notes:** • Audio-only content may be delivered as a file, as streamed file, or other means. • Other short sounds such as confirmation beeps and error notifications are not included in this requirement.
Related Standards	508 1194.22(a): Equivalent text descriptions WCAG2: 1.1.1 Non-text Content WCAG2: 1.2.1 Audio-only and Video-only (Prerecorded)
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find interface components that play audio-only content when activated. b. **[Web only]** Use the WAT (Doc Info - List of Multimedia files) to identify audio-only files. c. Find other audio content that plays automatically or upon activation of a control. **Notes:** • An audio-only file may be stored in a synchronized media format. For example, a speech is stored in a file where the video is simply a static graphic of the speaker's name and location. If the video component is static, and the information displayed in the video is also available as screen text, then treat the file as audio-only. • **[Web only]** Sometimes the WAT list of files will not work correctly. This can be due to scripted links. If the WAT list does not work, a manual inspection may be required to determine relevant files for testing.
Test Instruction 2: Inspecting/Using Components	a. Check that the transcript is accessible screen text (i.e., an image of a transcript with no ALT-text would not be sufficient to pass this test). b. Open the transcript and play the audio-only content. c. Check that the information in the transcript is an accurate and complete representation of the audio-only content. Note the inclusion or absence of relevant items in addition to dialogue, such as doors banging, sirens wailing and so forth.

Test Instruction 3a: Section 508 Failure Conditions	• Audio-only content is not accompanied by a transcript.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• Audio-only content is accompanied by a transcript that is not accurate or complete.
	  ○ Fails 1194.22(a): Equivalent text descriptions.

Test Instruction 3b: WCAG2 Failure Conditions	• Audio-only content is not accompanied by a transcript.
	  ○ Fails: 1.1.1 Non-text Content
	  ○ Fails 1.2.1 Audio-only and Video-only (Prerecorded)
	• Audio-only content is accompanied by a transcript that is not accurate or complete.
	  ○ Fails: 1.1.1 Non-text Content
	  ○ Fails 1.2.1 Audio-only and Video-only (Prerecorded)

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #18
	• Audio-only content has a transcript supplied AND the transcript is an accurate and complete representation of the audio-only content.
	  ○ Passes Baseline Requirement #18
	• There are no audio-only files.
	  ○ Not applicable (Baseline Requirement #18)

Advisory: Tips for streamlined test processes	• If audio is synchronized with video, slides, animations, or other time-based visual media, then use the synchronization test instead. • The proximity of the audio content to any control to reveal the transcript is covered by the focus (order) test (i.e., whether there is a logical order for content).

19. Video (descriptions)

Requirement	19. Video-only content must be accompanied by descriptions.
Rationale	For video-only content (e.g., animations of processes), some users will not be able to see the content (delivered as a video-only file, or other animation). Therefore, there needs to be an alternative delivery method for the information. The alternative description can be text, or an audio file describing what is being shown. **Notes:** • **[Web only]** If no description is supplied (either in text or in an audio file), the Section 508 failure defaults to a missing text description. • Other short animation effects such as button activation highlights and file shrink/disappear on closure are not included in this requirement.
Related Standards	508 1194.21(h): Animation 508 1194.22(a): Equivalent text descriptions 508 1194.24(d): Video descriptions WCAG2: 1.1.1 Non-text Content WCAG2: 1.2.1 Audio-only and Video-only (Prerecorded)
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find interface components that play video-only content when activated. b. **[Web only]** Use the WAT (Doc Info - List of Multimedia files) to identify video-only files. **Notes:** • A video-only file may be stored in a synchronized media format. For example, an animation is stored in a file where the audio is absent or can be considered incidental (e.g., background music that does not influence the comprehension of the animation). If the audio component is absent or incidental, then treat the file as video-only. • **[Web only]** Sometimes the WAT list of files will not work correctly. This can be due to scripted links. If the WAT list does not work, a manual inspection may be required to determine relevant files for testing.

Test Instruction 2: Inspecting/Using Components	a. Check that the description is available:
	  ○ as accessible screen text (i.e., an image of a description with no ALT-text would not be sufficient to pass this test), or
	  ○ as an audio file.
	b. Open the description and play the video-only content.
	c. Check that the information in the description is an accurate and complete representation of the video-only content.
	
	Note:
	• When accompanying a video-only file with an audio description file, the files do not have to be synchronized.

Test Instruction 3a: Section 508 Failure Conditions	• **[Web only]** A video-only file does not have a description.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** A video-only file has text descriptions that are not accurate or complete.
	  ○ Fails 1194.22(a): Equivalent text descriptions.
	• **[Web only]** A video-only file has audio descriptions that are not accurate or complete.
	  ○ Fails 1194.24(d): Video descriptions.
	• *[SW only]* An animation in SW does not have descriptions (text or audio).
	  ○ Fails 1194.21(h): Animation.
	• *[SW only]* An animation has descriptions (text or audio) that are not accurate or complete.
	  ○ Fails 1194.21(h): Animation.

Test Instruction 3b: WCAG2 Failure Conditions	• Video-only content does not have a description.
	  ○ Fails: 1.1.1 Non-text Content
	  ○ Fails 1.2.1 Audio-only and Video-only (Prerecorded)
	• Video-only content has text descriptions that are not accurate or complete OR audio descriptions that are not accurate or complete.
	  ○ Fails: 1.1.1 Non-text Content
	  ○ Fails 1.2.1 Audio-only and Video-only (Prerecorded)

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #19
	• Video-only content has descriptions supplied AND the descriptions are an accurate and complete representation of the video-only content.
	  ○ Passes Baseline Requirement #19
	• There is no video-only (/animation) content.
	  ○ Not applicable (Baseline Requirement #19)

Advisory: Tips for streamlined test processes	• If video is synchronized with audio, meaningful sounds, narration, or other time based visual media, then use the synchronization test instead. • The proximity of the video content to any control to reveal the description is covered by the focus (order) test (i.e., whether there is a logical order for content).

20. Synchronized media (captions)

Requirement	20. Synchronized media must have captions that are time-synchronized with the dialog and relevant sounds.
Rationale	Synchronized media is a presentation consisting of time-synchronized video and audio. Synchronized media includes public information films, Web casts, press conferences, and online training presentations. A prime consideration for synchronized media is that some users will not be able to hear the content well or hear it at all. Therefore there needs to be another mode to provide the audio information. This usually means captions (text showing what is being said, and other relevant sounds). Captions need to be available, but do not necessarily need to be turned on by default. For example, users who need captions can switch them on with a control (usually a 'CC' button for Closed Captions). If there is no means of switching modes, then the default mode must be accessible (i.e., Open Captions). Because captions must be time-synchronized, separate transcripts will not meet this requirement on their own.
Related Standards	508 1194.22(b): Synchronized Alternatives 508 1194.24(c): Captions WCAG2: 1.2.2 Captions (Prerecorded) WCAG2: 1.2.4 Captions (Live)
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find interface components that play synchronized media when activated. This includes streaming media, and streaming live events. b. **[Web only]** Use the WAT (Doc Info - List of Multimedia files) to find synchronized media files. **Notes:** • A synchronized media file may be used to store non-synchronized media format. For example, a speech is stored in a synchronized media file where the video is simply a static image of the speaker's face with a caption. If the video component is static, and the information displayed in the video is also available as screen text, then treat the file as audio-only rather than synchronized media. • **[Web only]** Sometimes the WAT list of files will not work correctly. This can be due to scripted links. If the WAT list does not work, a manual inspection may be required to determine relevant files for testing.
Test Instruction 2: Inspecting/Using Components	a. Enable the captioning for the synchronized media. b. Play the synchronized media content. c. Check that the information in the captions is an accurate, synchronized and complete representation of the dialogue and other relevant sounds in the synchronized media.

Test Instruction 3a: Section 508 Failure Conditions	• Synchronized media does not have captions.
	  ○ Fails 1194.24(c): Captions.
	• Synchronized media has captions that are not accurate or complete.
	  ○ Fails 1194.24(c): Captions.
	• Synchronized media has captions that are not synchronized with dialog and relevant sounds.
	  ○ Fails 1194.22(b): Synchronized Alternatives.

Test Instruction 3b: WCAG2 Failure Conditions	• Synchronized media does not have captions.
	  ○ Fails 1.2.2 Captions (Prerecorded)
	• Synchronized media has captions that are not accurate or complete.
	  ○ Fails 1.2.2 Captions (Prerecorded)
	• Synchronized media has captions that are not synchronized with dialog and relevant sounds.
	  ○ Fails 1.2.2 Captions (Prerecorded)
	• Captions are not provided for streaming of live media events.
	  ○ Fails 1.2.4 Captions (Live)

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #20
	• Synchronized media has captions AND the captions are an accurate, synchronized and complete representation of the audio contained in the synchronized media.
	  ○ Passes Baseline Requirement #20
	• There is no synchronized media.
	  ○ Not applicable (Baseline Requirement #20)

Advisory: Tips for streamlined test processes	• Testing synchronized media is different to testing audio-only content (test #18). • Testing synchronized captions AND synchronized descriptions at the same time may be more time effective, so long as both are given equal weight. • It is preferable to have the media on the main page for all users captioned and audio described, as current technology permits this. It is acceptable to have separate files for captioned and/or audio described versions. • Testing of synchronized media players is usually a software test of the plug-in.

21. Synchronized media (descriptions)

Requirement	21. Synchronized media must have audio descriptions that are time-synchronized with the video.
Rationale	Synchronized media is a presentation consisting of time-synchronized video and audio. Synchronized media includes public information films, Web casts, press conferences, and online training presentations. A prime consideration for synchronized media is that some users will not be able to see the content well or see it at all. Therefore there needs to be another mode to provide descriptions of the visual information. In synchronized media, this usually means additional narration inserted during breaks in the dialog, describing visual events and cues. Audio descriptions need to be available, but are not required to be turned on by default. For example, users who need descriptions can switch them on with a control. If there is no means of switching modes, then the audio descriptions must be enabled by default. The Alternative presentation of information must allow understanding of the relevant information. For example, descriptions might include the looks on people's faces, people handing items to each other, or who has entered the room. Synchronization is required for the Alternative presentation modes. Because descriptions must be synchronized, a text transcript will not meet this requirement. Synchronized media content cannot be played and then followed by a summary of the visual events. Instead, the visual events must be described as they are happening, usually during breaks in dialogue.
Related Standards	508 1194.22(b): Synchronized Alternatives 508 1194.24(d): Descriptions WCAG2: 1.2.3 Audio Description or Media Alternative (Prerecorded) WCAG2: 1.2.5 Audio Description (Prerecorded)
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Find interface components that play synchronized media when activated. b. **[Web only]** Use the WAT (Doc Info - List of Multimedia files) to find synchronized media files. **Notes:** • A synchronized media file may be used to store non-synchronized media format. For example, an animation is stored in a synchronized media file where the audio is absent or can be considered incidental (e.g., background music that does not influence the comprehension of the animation). If the audio component is absent or incidental, then treat the file as video-only. • **[Web only]** Sometimes the WAT list of files will not work correctly. This can be due to scripted links. If the WAT list does not work, a manual inspection may be required to determine relevant files for testing.

Test Instruction 2: Inspecting/Using Components	a. Enable the audio descriptions for the synchronized media. b. Play the synchronized media content. c. Check that the audible description is an accurate, synchronized and complete representation of the relevant visual events in the synchronized media.

Test Instruction 3a: Section 508 Failure Conditions	• Synchronized media is not audio described.
	  ○ Fails 1194.24(d): Descriptions.
	• Synchronized media is audio described, but the descriptions are not accurate or complete.
	  ○ Fails 1194.24(d): Descriptions.
	• Synchronized media is audio described, but the descriptions are not synchronized with video.
	  ○ Fails 1194.22(b): Synchronized Alternatives.

Test Instruction 3b: WCAG2 Failure Conditions	• Synchronized media is not audio described.
	  ○ Fails 1.2.3 Audio Description or Media Alternative (Prerecorded)
	  ○ Fails 1.2.5 Audio Description (Prerecorded)
	• Synchronized media is audio described, but the descriptions are not accurate or complete.
	  ○ Fails 1.2.3 Audio Description or Media Alternative (Prerecorded)
	  ○ Fails 1.2.5 Audio Description (Prerecorded)
	• Synchronized media is audio described, but the descriptions are not synchronized with video.
	  ○ Fails 1.2.3 Audio Description or Media Alternative (Prerecorded)
	  ○ Fails 1.2.5 Audio Description (Prerecorded)

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #21
	• Synchronized media is audio described AND the descriptions are an accurate, synchronized and complete representation of the video contained in the synchronized media.
	  ○ Passes Baseline Requirement #21
	• There is no synchronized media.
	  ○ Not applicable (Baseline Requirement #21)

Advisory: Tips for streamlined test processes	• Testing synchronized media is different from testing video-only content (test #19). • Testing synchronized captions AND synchronized descriptions at the same time may be more time effective, so long as both are given equal weight. • It is preferable to have the media on the main page for all users captioned and audio described, as current technology permits this. It is acceptable to have separate files for captioned and/or audio described versions. • Testing of synchronized media players is usually a software test of the plug-in.

22. Style-sheet non-dependence

Requirement	22. **[Web only]** Web pages must be structured so that their reading order is consistent whether they are viewed with or without an associated style sheet. Layout and appearance of the page may change visually, as long as the logical reading order is maintained.
Rationale	Style sheets are a means to provide visual formatting information to complement a Web page's content. The original intention behind style sheets was to separate presentation from content. The text, images, links etc. comprise the 'content', and things such as font choice, background color, link underlining etc. comprise the presentation 'style'. A Web page should in theory always be readable and functional without the developer's style sheet, since content is separate from presentation. However, it is possible for developers to inadvertently deliver content through style. For example, a background image can be applied with a style sheet, but if that background image also contains important information, such as an organization's name, logo and contact details, then content is no longer separate from presentation. Because of their particular visual needs, some people with visual impairments create their own style sheets (font color, background color, etc.) to replace the provided style sheet. When content is not properly separated from presentation, it becomes difficult or impossible to read the information on the screen. Therefore, pages must be tested with style sheet information removed, to ensure that all content is still being delivered to the user. For additional information/guidance, see also WCAG 2 glossary entry "Accessibility Supported"
Related Standards	508 1194.22(d): Readable without Style Sheets WCAG2: 1.1.1 Non-text content WCAG2: 1.3.2 Meaningful Sequence WCAG2: 1.3.3 Sensory Characteristics
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Look at the content on the original page to determine the logical order. b. Find hidden content for comparison purposes.
Test Instruction 2: Inspecting/Using Components	a. Use WAT (IE - Toggle CSS). Check for the following: ○ Do any meaningful images disappear (i.e., images that are set to show only with the style sheet) ○ Does the order of the content change from the logical order to a non-logical order? ○ Does any content become illegible due to overlapping? ○ Is hidden content from the original page still available? ○ Does any unintended content get introduced (i.e., content that never exists on the page but may be called up on other pages.)?

Test Instruction 3a: Section 508 Failure Conditions	• When CSS is disabled, meaningful images disappear
	  ○ Fails 1194.22(d): Readable Style Sheets.
	• When CSS is disabled, the content order is not logical
	  ○ Fails 1194.22(d): Readable Style Sheets.
	• When CSS is disabled, content becomes illegible.
	  ○ Fails 1194.22(d): Readable Style Sheets.
	• When CSS is disabled, content becomes unavailable.
	  ○ Fails 1194.22(d): Readable Style Sheets.
	• When CSS is disabled, unintended content shows on the page.
	  ○ Fails 1194.22(d): Readable Style Sheets.

Test Instruction 3b: WCAG2 Failure Conditions	• When CSS is disabled, meaningful images disappear
	  ○ Fails 1.1.1 Non-text content.
	• When CSS is disabled, the content order is not logical
	  ○ Fails 1.3.2 Meaningful Sequence
	  ○ Fails 1.3.3 Sensory Characteristics
	• When CSS is disabled, content becomes illegible.
	  ○ Fails 1.3.3 Sensory Characteristics
	• When CSS is disabled, content becomes unavailable.
	  ○ Fails 1.3.2 Meaningful Sequence
	• When CSS is disabled, unintended content shows on the page.
	  ○ Fails 1.3.2 Meaningful Sequence

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #22
	• When CSS is disabled, all meaningful images stay AND the order remains logical AND content remains legible AND content remains available AND unintended content does not show on the page.
	  ○ Passes Baseline Requirement #22
	• Style sheets are not used.
	  ○ Not applicable (Baseline Requirement #22)

Advisory: Tips for streamlined test processes	• It may be necessary to refresh the page (F5) a number of times to ensure that all pertinent CSS images are found. • A tester may find it easiest to toggle the style sheet view while testing or use two browser windows. • For content on the original page that is in layout tables, it is possible to produce a linearized representation that may be useful in determining whether a logical order is used. To linearize table content, use WAT (Tables - Linearize). This can be used to supplement the test, but is not part of the main test.

23. Frames

Requirement	23. **[Web only]** Frames and iframes must have a meaningful description using the title or name attribute.
Rationale	Frames are a means of separating out sections of a Web page into different navigable regions. To mouse users, the separation of a Web page into sections means that they can scroll the information in one frame without affecting another frame. Keyboard only users who are able to see can navigate between frames (F6 key is the browser default for this function). Non-visual users can also use the keyboard to navigate between frames, but if there is no programmatic name for the frames, the user has to read through the content of each frame in an attempt to discern where the information they need might be. This can take a long time, and can lead nonvisual users to make errors. For this reason, it is necessary for each frame to include a descriptive name. The name should make sense when spoken in isolation as the user navigates between frames.
Related Standards	508 1194.22(i): Descriptive Frame Titles WCAG2: 1.1.1 Non-text Content
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Use WAT (Select Frames – Frame Name / Title) to determine whether there are frames.
Test Instruction 2: Inspecting/Using Components	a. Use WAT (Select Frames – Frame Name / Title) to check each frame and iframe for a meaningful and unique content description in the Name or TITLE attribute.
Test Instruction 3a: Section 508 Failure Conditions	• A frame or iframe does not have a meaningful and unique title or Name. ○ Fails 1194.22(i): Descriptive Frame Titles.
Test Instruction 3b: WCAG2 Failure Conditions	• A frame or iframe does not have a meaningful and unique title or name. ○ Fails 1.1.1 Non-text Content

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #23
	• Each frame or iframe has a meaningful and unique title or name.
	  ○ Passes Baseline Requirement #23
	• There are no frames or iframes
	  ○ Not applicable (Baseline Requirement #23)

Advisory: Tips for streamlined test processes	• This test may be related to Page Titles (#12).

24. Alternate pages

Requirement	24. **[Web only]** When the primary page/site cannot be made accessible, an Alternative page/site must contain equivalent and up-to-date content.
Rationale	An ' Alternate Page' is an accessible version containing the same information as the primary page. Alternate pages will usually contain text in place of the inaccessible content from the primary page. For example, a complex organizational chart may be written in prose. The text must be equivalent, and it must be kept up-to-date. An ' Alternate Page' should only be provided for accessibility when the primary page cannot be made accessible. The accessible version must contain the same information as the primary page. **Note:** • The information should be 'equivalent', but by definition this is not going to be 'exactly the same'. The main points, themes, concepts etc. that the authors are trying to get across in the primary content should also come across in the alternate page. For example, if a complex chart on the primary page shows a year with a small increases in earnings in Q2 and a large decrease in Q2, and the text discusses why these trends seem to be occurring, the Alternate page should convey the trends, and the high and low data points of interest. An alternate page that just gave all the data points in linear form, with no highlighting of the trends under consideration, would not be considered equivalent.
Related Standards	508 1194.22(k): Text only or Alternative versions WCAG2: Conformance requirement #1: Conforming alternate version
Tools Necessary	None
Test Instruction 1: Finding Applicable Components	a. Determine whether there are any Alternate pages/sites by examining the content (pay particular attention to content containing maps, directions, complex charts etc.).
Test Instruction 2: Inspecting/Using Components	a. Compare the content of the primary page/site and the Alternate page/site, noting any information differences and/or out-of date material.
Test Instruction 3a: Section 508 Failure Conditions	• An Alternate page/site is provided, but the information is not equivalent to and up to date with the primary page/site. ○ Fails 1194.22(k): Text only or Alternative versions.
Test Instruction 3b: WCAG2 Failure Conditions	• An Alternate page/site is provided, but the information is not equivalent to and up to date with the primary page/site. ○ Fails Conformance requirement #1: Conforming alternate version

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #24
	• An alternate page/site contains equivalent, up-to-date information compared with the primary page/site.
	  ○ Passes Baseline Requirement #24
	• There are no alternate pages/sites.
	  ○ Not applicable (Baseline Requirement #24)

Advisory: Tips for streamlined test processes	• This is a test of equivalency of the information on an Alternate page/site. • The Alternate page must pass all relevant tests for accessibility. • The decision of whether to actually provide an Alternate page/site or not will rest with individual agencies and their policies. • Agencies may need to make policies on their definition of 'up-to-date' (i.e., immediately with any changes, within an hour, within a day etc.). • Historically, text-only versions of a Web page were employed because plug-ins and synchronized media were not accessible. Over time, the accessibility of plug-ins and the content they contain (e.g., electronic documents, forms, training courses in flash) has improved to the point where it is very rare that the primary page cannot be made accessible. However, maps and directions, and very complex diagrams and charts remain difficult to make accessible without Alternate pages.

25. Time outs

Requirement	25. Users of assistive technology must be alerted about a pending time out, and users must be able to request more time to complete their task.
Rationale	Messages and/or instructions to the user requesting their response within a given time are typically associated with sites that require a secure login. This includes both server time outs and client side security time outs. People who use AT such as screen reader software or voice input software may require more time than other users to assimilate the information and execute the controls on a Web page or software application. Because AT users may need more time, applications that have a time out must provide (a) prior notification/warning that a time out is about to occur, and (b) a means for the user to request more time. For additional information/guidance, see also WCAG 2 Understanding Guideline 2.2
Related Standards	508 1194.22(p): Time out notification WCAG2: 2.2.1 Timing Adjustable
Tools Necessary	Stopwatch.
Test Instruction 1: Finding Applicable Components	a. Determine if there is a timeout function from the application's documentation, or by leaving the session inactive for a period.
Test Instruction 2: Inspecting/Using Components	a. Check that an alert for a timeout is displayed, rather than the application exiting without warning. b. Check that the user is offered the option to continue their task (i.e., request more time), or exit. c. Check that the alert message is displayed for at least 20 seconds before the page actually times out. **Notes:** • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• A time-out occurs, and users are not alerted.
	  ○ Fails 1194.22(p): Time out notification.
	• A time-out occurs, and users cannot request more time.
	  ○ Fails 1194.22(p): Time out notification.
	• A time-out occurs, and is displayed for less than 20 seconds.
	  ○ Fails 1194.22(p): Time out notification.

Test Instruction 3b: WCAG2 Failure Conditions	• A time-out occurs, and users are not alerted.
	  ○ Fails 2.2.1 Timing Adjustable
	• A time-out occurs, and users cannot request more time.
	  ○ Fails 2.2.1 Timing Adjustable
	• A time-out occurs, and is displayed for less than 20 seconds.
	  ○ Fails 2.2.1 Timing Adjustable

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #25
	• A time-out occurs, users are alerted, AND users can request more time, AND the length of time that the alert is displayed is 20 seconds or more.
	  ○ Passes Baseline Requirement #25
	• There is no time out function.
	  ○ Not applicable (Baseline Requirement #25)

Advisory: Tips for streamlined test processes	• Remind testers that when the time-out occurs, visible focus should shift to the time-out alert. • In some cases, it may be necessary to contact the application authors to clarify the conditions under which time-outs occur. • Security policies at a given agency may require certain systems to time-out less than 20 seconds after the alert is first displayed. If security policies do override this requirement (via 'undue burden' tests, for example), the time should still be reasonable enough for the AT to user to read through and navigate to their choice (e.g., 'continue' or 'exit'). Additional testing with AT (screen readers, speech recognition etc.) may be needed to determine whether the time is considered reasonable.

26. Image maps

Requirement	26. **[Web only]** Server-side image maps may not be used. Client-side image-maps must be used instead.
Rationale	Server-side image maps are not keyboard accessible and incompatible with assistive technologies used by people with disabilities. Any server-side image maps must therefore be replaced by client-side ones.
Related Standards	508 1194.22(e) Redundant text links on server-side image maps 508 1194.22(f) Client side not server side WCAG 2: Conformance requirement #4: Only accessibility ways
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Use WAT (Images - Show Image Maps) to determine whether there are image maps in use.
Test Instruction 2: Inspecting/Using Components	a. Use WAT (Images - Show Image Maps) to list the types of image map in use.

Test Instruction 3a: Section 508 Failure Conditions	• A server-side image map is in use.
	  ○ Fails 1194.22(e) Redundant text links on server-side image maps.
	  ○ Fails 1194.22(f) Client side not server side.
	Note:
	• The presence of any server-side image is technically a failure of 1194.22(f). Because this standard has been interpreted to mean that server-side image maps must be replaced by client-side image maps, 1194.22(e) becomes an automatic failure when server-side images are present.

Test Instruction 3b: WCAG2 Failure Conditions	• A server-side image map is in use. ○ Fails Conformance requirement #4: Only accessibility ways

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #26
	• There are no server-side image maps (only client-side image maps are used).
	  ○ Passes Baseline Requirement #26
	• There are no image maps.
	  ○ Not applicable (Baseline Requirement #26)

Advisory: Tips for streamlined test processes	• This test is only for the type of image maps used. Server-side images are rare. All other aspects of image maps (use with keyboard, link names, ALT-text names etc.) are covered by other tests.

27. Plug-in Links

Requirement	27. **[Web only]** When public-facing pages utilize content delivered via plug-ins, or contain downloadable content that must be opened with a separate application, a link to obtain the plug-in and/or application must be provided.
Rationale	It may be necessary or desirable to deliver content that must be displayed in a separate application, or via a browser plug-in, such as PDF and other electronic document files, and synchronized media. For public facing Web pages, there must be a link provided either directly (i.e., next to the content) or indirectly (i.e., a page providing links to all plug-ins used on a site). **Notes:** • Most interfaces will notify users that a plug-in is missing, and will help the user find and install the necessary plug-in. Some plug-ins do not do this, which is why this requirement remains in place. • Agencies usually restrict the software that can be installed on users' computers, and will also provide commonly used plug-ins as part of their enterprise architecture. For this reason, it is not required to provide a link to plug-ins for non-public facing (i.e., intranet) pages.
Related Standards	508 1194.22(m): Plug-ins
Tools Necessary	WAT
Test Instruction 1: Finding Applicable Components	a. Use WAT (Images - Show Images). If non-HTML images are used, WAT will not mark up the code (i.e., there will be no alerts given for a lack of ALT-text). Open the context menu (right click) on the image to determine the type of file in use. b. Use the WAT (Doc Info - List of Downloadable files) to identify files that must be viewed with a separate application. **Notes:** • Sometimes the WAT list of files will not work correctly. This can be due to scripted links. If the WAT list does not work, a manual inspection may be required to determine relevant files for testing.
Test Instruction 2: Inspecting/Using Components	a. Use WAT (Structure - Show Other Elements - "a") to highlight all links on the page, or WAT (Doc Info - List Links). Determine whether there are links to the required plug-ins. **Notes:** • If the WAT command does not work or is unavailable, find all links on a page by TABbing through the page content. • Some images may contain links.
Test Instruction 3a: Section 508 Failure Conditions	• A link is not provided for required plug-ins on public-facing pages. ○ Fails 1194.22(m): Plug-ins.
Test Instruction 3b: WCAG2 Failure Conditions	• Not applicable.

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #27
	• Links are provided for required plug-ins on public-facing pages.
	  ○ Passes Baseline Requirement #27
	• There are no required plug-ins OR this is not a public-facing page.
	  ○ Not applicable (Baseline Requirement #27)

Advisory: Tips for streamlined test processes	• This is a test of whether there is a link to get the plug-in / application from a public-facing site. • The links to the plug-ins must pass the relevant baseline test for links. • The plug-in baseline test is a web test for a link to player software to access the plug-in's content. However, if the plug-in's content is software, the plug-in itself would need to be subjected to all relevant tests for software. • Agencies may need to make policies on which plug-ins to allow and use, and whether to require links for plug-ins on internal (intranet) sites.

28. Built-in accessibility features

Requirement	28. *[SW only]* Operating System (OS) user-configurable accessibility appearance settings and functions must not be disrupted or disabled by the software application.
Rationale	It is possible to write software that controls various aspects of the OS. The control commands may inadvertently cause an OS accessibility feature to deactivate. For example, Sticky Keys is a feature that enables users to use one finger or pointer to use the control key, the alt key, and the shift key sequentially (rather than the simultaneously, as they are ordinarily used). If a developer wanted to reset the keyboard state because they wanted to turn off the CAPS LOCK indicator, they must take care not to reset accessibility features at the same time. It is also possible to override OS accessibility appearance features. For example, in High Contrast mode, the color settings of standard windows components are modified throughout the OS. If a developer wanted to make a special green and brown camouflaged theme for his or her application, they would specify the exact colors that would be used in the menus and other window control components. By specifying the exact colors, rather than adopting system colors, they override the ability of the user to employ the high contrast settings that they need to access the application. In this example, an aesthetic preference results in non-compliance with the requirement not to interfere with the user's appearance settings. The accessibility features of Windows 7, Windows 8.1, and 10 (the platforms for which the baseline tests are written) contain the following user-configurable accessibility features that should not be disabled or disrupted by the software application: • All settings in the Ease of Access control panel • System color settings, including high contrast modes • System text size settings **Note:** • This requirement also applies to software that is embedded in a page displayed in a Web browser.
Related Standards	508 1194.21(b) Built-in Accessibility Features 508 1194.21(g): OS Individual display attributes 508 194.31(f): Use with physical limitations. 508 1194.31(c): Use without hearing. WCAG2: 1.4.4 Resize text WCAG 2: Conformance requirement #5: Non-interference
Tools Necessary	None

Test Instruction 1: Finding Applicable Components	a. Close the software application under test b. Set the system text size to 200%: **<u>If the software is embedded in a Web page:</u>** o Set the browser zoom control to 200% of normal size. **<u>If the software is running as a stand-alone application:</u>** o Open the Windows Ease of Access center in the control panel (Windows Key + U) o Choose ‘Make the computer easier to see', then 'Change the size of text and icons'. o Windows 7: 'Set custom text size (DPI)' and set to 200%. o Windows 8.1/10: In the ‘Change only the text size’ section, select Menu, then select a font size twice the current size. Do the same for title bars, message boxes, icons, tooltips, and palette titles. o Log out and back in when prompted. c. Set the display to High Contrast Black: press Left Alt + Left Shift + Print Screen keys. d. Set Sticky Keys to on: press the left shift key 5 times. An icon will show in the taskbar notification area to show that Sticky Keys is on. e. Set Sound Sentry to on: o Open the Windows Ease of Access center in the control panel (Windows Key + U). o Choose 'Use text or visual alternatives for sound', then 'Turn on visual notification for sounds (Sound Sentry). f. Restart the software application under test.
Test Instruction 2: Inspecting/Using Components	a. Check that the application adopted the high contrast appearance b. Check and that the text size has been increased, and that information is not cut off because of the larger font size (scrolling may be necessary, and is acceptable). c. Verify that Sticky Keys has not been disrupted in the OS (the icon in the taskbar notification area should be showing that Sticky Keys is still on). d. Verify that Sticky Keys has not been disrupted in the application. Find any text field in the application, and type a mix of upper and lower case characters using one finger only. e. Verify that Sound Sentry has not been disrupted in the OS. Open Notepad and type Ctrl+L. The Sound Sentry indicator should flash. **<u>Note:</u>** • After this test is complete, reset the OS Accessibility features (and browser settings, if applicable) to their default settings, and restart the application. • Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content.

Test Instruction 3a: Section 508 Failure Conditions	• User OS Color settings overridden by the software application.
	  ○ Fails 1194.21(g): OS Individual display attributes.
	• Text of application did not enlarge, or became illegible when enlarged.
	  ○ Fails 1194.21(g): OS Individual display attributes.
	• Sticky Keys functionality was disrupted in the application.
	  ○ Fails 1194.31(f) Use with physical limitations.
	• Sound Sentry functionality was disrupted by the application.
	  ○ Fails 1194.31(c) Use without hearing.
	• OS accessibility features (High Contrast, Sticky Keys, Sound Sentry and/or system text size (stand-alone software only)) were disrupted by the application.
	  ○ Fails 1194.21(b) Built-in Accessibility Features.

Test Instruction 3b: WCAG2 Failure Conditions	• User OS Color settings overridden by the software application.
	  ○ Fails Conformance requirement #5: Non-interference
	• Text size settings disrupted by the software application.
	  ○ Fails 1.4.4 Resize text
	  ○ Fails Conformance requirement #5: Non-interference
	• OS accessibility features disrupted by the software application.
	  ○ Fails Conformance requirement #5: Non-interference
	• Sticky Keys functionality disrupted in the application.
	  ○ Fails Conformance requirement #5: Non-interference.

Test Instruction 3c: Baseline Requirement Test Results	• Any failure in 3a
	  ○ Fails Baseline Requirement #28
	• User OS color settings, OS text size, and OS accessibility features are not overridden or disrupted by the software application.
	  ○ Passes Baseline Requirement #28

Advisory: Tips for streamlined test processes	• There are other accessibility features in Windows that could be relevant to test depending on the content of the application. For example, for an application that uses sounds a great deal, Sound Sentry may be worth testing. A failure of any other Windows accessibility functions would not constitute a failure of the baseline test.

Attachment A - Cross-Reference Tables

Note:
The names for Section 508 tests are provided as short-hand for reference in the tables that follow. These are not the official names. For the text of the actual standards see the original document.[1 (page 4)]

Baseline tests (cross-reference table)

No.	Baseline test	Scope	Section 508 coverage	WCAG 2 (reference only)
1.	Keyboard navigation	Both	21 SW (a): Keyboard Accessibility	1.3.1 Info and relationships 2.1.1 Keyboard 2.1.2 No Keyboard Trap
2.	Focus (visible)	Both	21 SW (c): Visual Focus	2.4.7 Focus Visible
3.	Focus (order)	Both	31 FPC (a): Use without vision 31 FPC (b): Use with low vision	2.4.3 Focus Order 3.2.3 Consistent Navigation
4.	Focus (Revealing hidden content)	Both	31 FPC (a): Use without vision 31 FPC (b): Use with low vision	2.4.3 Focus Order 3.2.2 On Input
5.	Skip-navigation links	**Web only**	22 Web (o): Method to Skip Repetitive Links	2.4.1 Bypass Blocks
6.	Multi-state components	Both	21 SW (d): Name, Role, State 31 FPC (a): Use without vision 31 FPC (b): Use with low vision	1.3.1 Info and Relationships 3.2.1 On Focus 3.2.2 On Input 4.1.2 Name, Role, Value
7.	Images	Both	21 SW (d): Name, Role, State 21 SW (e): Bitmap images 22 Web (a): Equivalent text descriptions	1.1.1 Non-text Content 3.2.4 Consistent Identification
8.	Color (meaning)	Both	21 SW (i): No color dependence to convey information 22 Web (c): No color dependence to convey information	1.1.1 Non-text Content 1.4.1 Use of Color
9.	Color (contrast)	Both	31 FPC (b): Use with low vision	1.4.3 Contrast (Minimum)
10.	Flashing (reserved)	Both	21 SW (k): Blinking objects 22 Web (j): No flickering Interface components.	2.3.1 Three flashes or below threshold
11.	Forms (associated instructions)	Both	21 SW (f): Input text 21 SW (l): Forms 22 Web (n): Labels for forms	1.3.1 Info and Relationships 3.3.2 Labels or Instructions
12.	Page Titles	Both	31 FPC (a): Use without vision 31 FPC (b): Use with low vision	2.4.2 Page Titled
13.	Data Tables (headers)	Both	21 SW (d): Name, Role, State 22 Web (g): Identify row and column headers	1.3.1 Info and Relationships
14.	Data Tables (cell-header mapping)	Both	21 SW (d): Name, Role, State 22 Web (h): Associate Data with Headers	1.3.1 Info and Relationships
15.	Headings	**Web only**	31 FPC (a): Use without vision 31 FPC (b): Use with low vision	1.3.1 Info and Relationships

No.	Baseline test	Scope	Section 508 coverage	WCAG 2 (reference only)
16.	Links and user controls	Both	21 SW (d): Name, Role, State 22 Web (l): Functional Text for Scripts 31 FPC (a): Use without vision 31 FPC (b): Use with low vision	2.4.4 Link Purpose (In Context):
17.	Language	**Web only**	31 FPC (a): Use without vision 31 FPC (b): Use with low vision	3.1.1 Language of Page 3.1.2 Language of Parts
18.	Audio (transcripts)	Both	22 Web (a): Equivalent text descriptions	1.1.1 Non-text Content 1.2.1 Audio-only and Video-only (Prerecorded)
19.	Video (descriptions)	Both	22 Web (a): Equivalent text descriptions 21 SW (h): Animation 24 Multimedia (d): Video descriptions	1.1.1 Non-text Content 1.2.1 Audio-only and Video-only (Prerecorded)
20.	Synchronized media (captions)	Both	22 Web (b): Synchronized Alternatives 24 Multimedia (c): Captions	1.2.2 Captions (Prerecorded) 1.2.4 Captions (Live)
21.	Synchronized media (descriptions)	Both	22 Web (b): Synchronized Alternatives 24 Multimedia (d): Descriptions	1.2.3 Audio Description or Media Alternative (Prerecorded) 1.2.5 Audio Description (Prerecorded)
22.	Style-sheet non-dependence	**Web only**	22 Web (d): Readable Style Sheets	1.1.1 Non-text Content 1.3.2 Meaningful Sequence 1.3.3 Sensory Characteristics
23.	Frames	**Web only**	22 Web (i): Descriptive Frame Titles	1.1.1 Non-text Content
24.	Alternate pages	**Web only**	22 Web (k): Text only or Alternative versions	Conformance requirement #1: conforming Alternate version
25.	Time outs	Both	22 Web (p): Time out notification	2.2.1 Timing Adjustable
26.	Image maps	**Web only**	22 Web (e) Redundant text links on server-side image maps[17] 22 Web (f) Client side not server side	Conformance requirement #4: only accessibility ways
27.	Plug-in Links	**Web only**[18]	22 Web (m): Plug-ins	Not Applicable.

[17] If Web (f) is a failure, Web (e) is an automatic failure. See test notes for details.

[18] The plug-in baseline test is a web test for a link to player software to access the plug-in's content. However, if the plug-in's content is software, the plug-in itself would need to be subjected to all relevant tests for software.

No.	Baseline test	Scope	Section 508 coverage	WCAG 2 (reference only)
28.	Built-in accessibility features	*SW only*	21 SW (b) Built-in Accessibility Features 21 SW (g): OS Individual display attributes 31 FPC (c): Use without hearing 31 FPC (f): Use with physical limitations	1.4.4 Resize text Conformance requirement #5: non-interference

Section 508 (cross-reference table)

Para.	Name	Baseline test	Scope
21 SW (a)	Keyboard Accessibility	1. Keyboard navigation	Both
21 SW (b)	Built-in Accessibility Features	28. Built-in accessibility features	*SW only*
21 SW (c)	Visual Focus	2. Focus (visible) 4. Focus (Revealing hidden content)	Both
21 SW (d)	Name, Role, State	6. Multi-state components 7. Images 13. Data Tables (headers) 14. Data Tables (cell-header mapping) 16. Links and user controls	Both
21 SW (e)	Bitmap images	7. Images	Both
21 SW (f)	Input text	11. Forms (associated instructions)	Both
21 SW (g)	OS Individual display attributes	28. Built-in accessibility features	*SW only*
21 SW (h)	Animation	19. Video (descriptions)	Both
21 SW (i)	No color dependence to convey information	8. Color (meaning)	Both
21 SW (j)	Variety of color selections	Not applicable *(see note below)*	*N/A*
21 SW (k)	Blinking objects	10. Flashing (reserved)	Both
21 SW (l)	Forms	11. Forms (associated instructions)	Both
22 Web (a)	Equivalent text descriptions	7. Images 18. Audio (transcripts) 19. Video (descriptions)	Both
22 Web (b)	Synchronized Alternatives	20. Synchronized media (captions) 21. Synchronized media (descriptions)	Both
22 Web (c)	No color dependence to convey information	8. Color (meaning)	Both
22 Web (d)	Readable Style Sheets	22. Style-sheet non-dependence	**Web only**
22 Web (e)	Redundant text links on server-side image maps	26. Image maps	**Web only**
22 Web (f)	Client side not server side	26. Image maps	**Web only**
22 Web (g)	Identify row and column headers	13. Data Tables (headers)	Both
22 Web (h)	Associate Data with Headers	14. Data Tables (cell-header mapping)	Both
22 Web (i)	Descriptive Frame Titles	23. Frames	**Web only**
22 Web (j)	No flickering Interface components.	10. Flashing (reserved)	Both
22 Web (k)	Text only or Alternative versions	24. Alternate pages	**Web only**
22 Web (l)	Functional Text for Scripts	16. Links and user controls	Both
22 Web (m)	Plug-ins	27. Plug-in Links	**Web only**

Para.	Name	Baseline test	Scope
22 Web (n)	Labels for forms	11. Forms (associated instructions)	Both
22 Web (o)	Method to Skip Repetitive Links	5. Skip-navigation links	**Web only**
22 Web (p)	Time out notification	25. Time outs	Both
24 Multimedia (c)	Captions	20. Synchronized media (captions)	Both
24 Multimedia (d)	Video descriptions	19. Video (descriptions) 21. Synchronized media (descriptions)	Both
31 FPC (a)	Use without vision	4. Focus (Revealing hidden content) 6. Multi-state components 12. Page Titles 15. Headings 16. Links and user controls 17. Language (Web only)	Both
31 FPC (b)	Use with low vision	4. Focus (Revealing hidden content) 6. Multi-state components 9. Color (contrast) 12. Page Titles 15. Headings 16. Links and user controls 17. Language (Web only)	Both
31 FPC	Use without hearing	28. Built-in accessibility features	*SW only*
31 FPC (f)	Use with physical limitations	28. Built-in accessibility features	*SW only*

Section 508 standards not included in this baseline:

The Baseline tests include instructions and failure conditions for all Section 508 standards except the following:

Subpart B - Technical Standards

1194.21 Software applications and operating systems

(j) Variety of color selections

Comment: *SW (g) requires that applications do not override user selected color and contrast selections. Passing SW (g) means that the user-selectable ranges of colors in the OS (Microsoft Windows for this baseline test) are available. Because the baseline has standardized on Windows, SW (j) is automatically met through SW (g) in Baseline #28. Agency-specific tests may be developed for SW(j) if applications do not meet SW(g).*

1194.22 ***Web-based*** intranet and internet information and applications

(e) Redundant text links on server-side image maps

Comment: *Web (f) has been interpreted to mean that server-side image maps must be replaced by client-side image maps. In Baseline #26, , Web (e) becomes an automatic failure where server-side image maps exist.*

1194.23 Telecommunications products (all)

Comment: *Telecom products are converging with software and Web browsing capabilities. However, at this time the baseline tests herein cover use of software and Web sites on PCs (i.e., desktops and laptops).*

1194.24 Video and multimedia products[19]

(a) Television tuners and captioning

(b) Television tuners and secondary audio program

Comment: *The baseline tests herein cover media displayed on PCs only.*

(e) User selection of Alternatives

Comment: *This standard is already covered by the baseline tests.*

1194.25 Self-contained, closed products (all)

Comment: *The baseline tests cover software and Web sites running on desktops and laptops, not on public kiosks and similar devices.*

1194.26 Desktop and portable computers (all)

Comment: *The baseline tests cover software and Web sites running on desktops and laptops, but not the PC hardware itself.*

Subpart C - Functional Performance Criteria

508 1194.31 (d) Use with limited hearing

Comment: *The main requirements regarding hearing are already covered by the Web and multimedia baseline tests. Short sounds such as confirmation beeps and error notifications are not included in the Baseline tests.*

508 1194.31 (e) Use without speech

Comment: *The main requirements regarding use without speech are already covered by the software and Web baseline tests.*

Subpart D - Information, Documentation, and Support (all)

Comment: *Printed software documentation, if it is supplied with the product, is subject to this standard. However, the baseline tests are for the software itself.*

[19] The baseline includes the multimedia standards 1194.24(c) Captions, and 1194.24(d): Descriptions. It could be argued that these are already covered by the web standard 1124.22(b) which reads: *"Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation".* The problem is that 22(b) does not clearly describe what the "equivalent alternatives" are. 22(b) does, however, use the term "multimedia" (i.e., 1194.24). Thus, the interpretation is that the intended "equivalent alternatives" are captions and descriptions (i.e., 24(c) and 24(d)).

WCAG 2.0 (cross-reference table)

Note:
The following table is for reference only. The baseline tests align with, but do not necessarily cover WCAG 2.0 completely. Following the baseline tests should not be considered equitable to WCAG conformance. [4 (page 6)]

No.	Name	Baseline test	Scope
1.1.1	Non-text Content	7. Images 8. Color (meaning) 19. Video (descriptions) 18. Audio (transcripts) 22. Style-sheet non-dependence 23. Frames	Both
1.2.1	Audio-only and Video-only (Prerecorded)	18. Audio (transcripts) 19. Video (descriptions)	
1.2.2	Captions (Prerecorded)	20. Synchronized media (captions)	Both
1.2.3	Audio Description or Media Alternative (Prerecorded)	21. Synchronized media (descriptions) 19. Video (descriptions)	Both
1.2.4	Captions (Live)	20. Synchronized media (captions)	Both
1.2.5	Audio Description (Prerecorded)	21. Synchronized media (descriptions)	Both
1.3.1	Info and Relationships	1. Keyboard navigation 6. Multi-state components 11. Forms (associated instructions) 13. Data Tables (headers) 14. Data Tables (cell-header mapping) 15. Headings	Both
1.3.2	Meaningful Sequence	22. Style-sheet non-dependence	**Web only**
1.3.3	Sensory Characteristics	22. Style-sheet non-dependence	**Web only**
1.4.1	Use of Color	8. Color (meaning)	Both
1.4.3	Contrast (Minimum)	9. Color (contrast)	Both
1.4.4	Resize text	28. Built-in accessibility features	Both
2.1.1	Keyboard	1. Keyboard navigation	Both
2.1.2	No Keyboard Trap	1. Keyboard navigation	Both
2.2.1	Timing Adjustable	25. Time outs	Both
2.3.1	Three flashes or below threshold	10. Flashing (reserved)	Both
2.4.1	Bypass Blocks	5. Skip-navigation links	**Web only**
2.4.2	Page Titled	12. Page Titles	Both
2.4.3	Focus Order	3. Focus (order) 4. Focus (Revealing hidden content)	Both
2.4.4	Link Purpose (In Context):	16. Links	Both

No.	Name	Baseline test	Scope
2.4.7	Focus Visible	2. Focus (visible)	Both
3.1.1	Language of Page	17. Language	**Web only**
3.1.2	Language of Parts	17. Language	**Web only**
3.2.1	On Focus	6. Multi-state components	Both
3.2.2	On Input	4. Focus (Revealing hidden content) 6. Multi-state components	Both
3.2.3	Consistent Navigation	3. Focus (order)	Both
3.3.2	Labels or Instructions	11. Forms (associated instructions)	Both
3.2.4	Consistent Identification	7. Images	Both
4.1.2	Name, Role, Value	6. Multi-state components	Both
N/A	Conformance requirement #1: conforming Alternate version	24. Alternate pages	**Web only**
N/A	Conformance requirement #4: only accessibility ways	26. Image maps	**Web only**
N/A	Conformance requirement #5: non-interference	28. Built-in accessibility features	*SW only*

WCAG 2.0 Success Criteria Not covered in Baseline Tests

AAA Success criteria (all)

> ***Comment:*** *The baseline tests herein are aligned with the WCAG Level A and Level AA success criteria. WCAG comments on the more stringent AAA:*
>
> > *"It is not recommended that Level AAA conformance be required as a general policy for entire sites because it is not possible to satisfy all Level AAA Success Criteria for some content."* [3 (page 6)]

1.4.2 Audio Control

2.2.2 Pause, Stop, Hide

2.4.5 Multiple Ways

2.4.6 Headings and Labels

3.3.1 Error Identification

3.3.4 Error Prevention (Legal, Financial, Data)

4.1.1 Parsing

> ***Comment:*** *The above success criteria do not map to any current Section 508 standard.*

3.3.3 Error Suggestion

> ***Comment:*** *Errors are already covered by baseline tests for keyboard access, focus, labels etc.*

1.4.5. Images of Text

> ***Comment:*** *The mapping of the coverage of this success criteria to 508 21a (Equivalent test descriptions) was better served by WCAG 1.1.1 (Non-text Content).*

Conformance Requirement #2: F*ull Pages*

Conformance Requirement #3: Complete Process

> ***Comment:*** *The above requirements are considered a given in Section 508 tests (i.e., the complete Web site must be compliant).*

Baseline tests not mapped to WCAG 2.0

28. Plug-in Links

> ***Comment:*** *The above tests is required for Section 508 compliance, but has no equivalent in WCAG 2.0 AA success criteria or conformance requirements.*

Attachment B - Flashing content test advisory notes

Agencies must include an evaluation of flashing/blinking content in their test processes. However, as of the publication of the current version of baseline tests, there is no agreed-upon testing method. The test number 10 is reserved for a future version of this document when an agreed-upon test process will be included. The following are advisory notes relating to tests of flashing content.

Why to include a flashing content test in a test process

Even though there is no baseline, there are two primary reasons to include a test: the Section 508 law, and the risk of injury to users.

The Section 508 standards require:

> § 1194.21Software applications and operating systems (k) Software shall not use flashing or blinking text, objects, or other components having a flash or blink frequency greater than 2 Hz and lower than 55 Hz.
>
> § 1194.22Web-based intranet and internet information and applications. (j) Pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz.

The standards are in place as an attempt to reduce the likelihood of causing a seizure in a user with photosensitive epilepsy. It is therefore incumbent on agencies to apply due diligence to try to lower the likelihood of causing injury.

Note:
WCAG 2.0 also includes two related success criteria:

> 2.2.2 Pause, Stop, Hide
>
> 2.3.1 Three Flashes or Below Threshold

The WCAG 2.0 Web site contains advice, commentary, and links to further information relating to the above success criteria that may be useful to consult when developing a streamlined test process.[3 (page 6)]

Why there is no baseline test for flashing

Despite exhaustive analysis efforts of DHS and SSA staff, a reliable, repeatable method to determine the number of flashes or blinks per second could not be found or established at the time of publication. There are many candidate methods to try, and it may be possible to create a software tool that can be accepted in the future. Candidate methods that were studied included:

- Seeking the code from developers to show the programmed cycles per second. This test is considered as too advanced for most testers (cycle values have to be translated through formulas to get a Hz value). Further, other program and operating system functions can slow or speed up a programmed value to something that differs from the intended value (in our analysis, the majority showed flash rates that differed to the code).
- A tester visually following the flashing, along with some counting aid (counting in the head "one one thousand two one thousand" etc., using a stopwatch or countdown

timer, using a metronome, and other methods). Each test involving human perception has its limitations and brings up inter-tester and intra-tester reliability questions.

- Using a software tool to blink at a known rate and placing it next to the flashing content to visually compare rates. Although this was promising, the ability of users varied in their capability of making measurements. After about 2.5 Hz, the testers could not reliably track both flashing objects. Further, getting the tool to blink at the desired rate on different computers was problematic.
- Using a software tool to capture and analyze the content displayed on screen. The tools proved unreliable, in part due to the mismatch between sampling frequency and the screen refresh rate. Interference can occur when flashes are in the process of being 'drawn' on the screen at the same time as the sampling is taking place.
- Using a video camera to capture the screen. This is considered a cumbersome test for general use, and it is subject to the same interference problems as with the screen capture software.

Requirement, and draft rationale

Requirement

Sections(s) of the screen should not flash at or above 3Hz.

Note:

Section 508 sets limits at 2Hz, but WCAG, produced later than Section 508, revises that figure to 3Hz based on research. It is likely that the Section 508 refresh will adopt the 3Hz figure, and so that requirement is adopted in the baseline.

Rationale

The following is advisory only. It will be finalized in future versions of this document when an agreed-upon test process is released.

A component that flashes or blinks in the visual field can cause adverse reactions in people who have photosensitive epilepsy. The size, intensity and duration that causes seizures varies from individual to individual. However, it is well established that objects flickering in the frequency range from 3Hz to 55Hz (from three times to 55 times per second) should be avoided.

Notes:

Scrolling ('marquee') text may cause a flashing effect under certain circumstances.

At flash rates approaching and above 55Hz, flashing can be imperceptible to the naked eye (the component(s) will look like they have a steady state). For this reason there is no test that deals with the higher cut-off point of 55Hz.

How to report on flashing content

When developing test processes, and reporting results from such test processes, agencies must include a test related to flashing, even though there is no baseline test.

- Results of tests should indicate the test method used.
- Results of tests for flashing can be accepted by other agencies at their discretion.
- Agencies who adopt the baseline tests and share results with one another cannot reject another agency's test results just because they do not accept the methods for testing

flashing content. Agencies can reject the flashing results, but will accept the remainder of the test results, until a reliable baseline test is chosen.

Attachment C - Baseline Test Report Checklists

Instructions

- Include one of the following checklists when sharing results between agencies:
  - Software-only test
  - Web-only test
  - Web+Software test (use for instances such as a web site that uses embedded flash content, or software that uses a web interface for its' user guide)
- The checklist should be placed at or near the front of the report.
- Include a short summary of the findings from the main report.
- All applicable lines should be marked in both tables (the baseline requirements and the Section 508 standards).
- For directions on which column to mark (Fail, Pass, N/A) see the reporting instructions in each baseline test, or the instructions in the streamlined test you are working from.
- Attach to the checklist a summary table explaining each baseline failure (the following are recommended minimum reporting requirements; additional supporting information may also be provided):
  - Enter the Baseline number (from the checklist)
  - Enter the Applicable requirement's short title (from the checklist)
  - Briefly describe what issues failed (e.g., "Missing Alt Text on Images"), and where the failure occurred (e.g., "Login screen", "Help Menu", "Multiple locations")
  - Add any additional notes on the test results (e.g., AT compatibility; potential work-arounds that might mitigate a failure, impact this failure may have on users with disabilities). If a failure applies to more than one baseline requirement, it can be mentioned here (it is not necessary to list the same failure multiple times).
- Include, attach, or reference the location of the full test report.

Note:
For general reporting guidance, see the section "*Developing a streamlined test process from this baseline*".

Software-only test

Section 508 Baseline Test Report Checklist

Include this checklist when sharing results between agencies

Agency: _____ Agency contact details:_____

Product name: _____ Version # _____ Date tested: _____

Summary of main findings:

_____

_____

_____

#	Applicable Baseline requirements	Fail	Pass	N/A
1.	Keyboard navigation	☐	☐	
2.	Focus (visible)	☐	☐	
3.	Focus (order)	☐	☐	
4.	Focus (Revealing hidden content)	☐	☐	☐
5.	Skip-navigation links	☐	☐	☐
6.	Multi-state components	☐	☐	☐
7.	Images	☐	☐	☐
8.	Color (meaning)	☐	☐	☐
9.	Color (contrast)	☐	☐	
10.	Flashing (reserved)*	(☐)	(☐)	☐
11.	Forms (associated instructions)	☐	☐	☐
12.	Page Titles	☐	☐	
13.	Data Tables (headers)	☐	☐	☐
14.	Data Tables (cell-header mapping)	☐	☐	☐
16.	Links and user controls	☐	☐	☐
18.	Audio (transcripts)	☐	☐	☐
19.	Video (descriptions)	☐	☐	☐
20.	Synchronized Media (captions)	☐	☐	☐
21.	Synchronized Media (descriptions)	☐	☐	☐
24.	Alternate pages	☐	☐	☐
25.	Time outs	☐	☐	☐
28.	Built-in accessibility features	☐	☐	☐

* *Flashing must be tested, but there is no agreed-upon baseline test. Include evaluation methods in attached test report.*

Para.	Applicable 508 Std.	Fail
21(a)	Keyboard Accessibility	☐
21(b)	Built-in Acc. Features	☐
21(c)	Visual Focus	☐
21(d)	Name, Role, State...	☐
21(e)	Bitmap images	☐
21(f)	Input text	☐
21(g)	Color & Contrast	☐
21(h)	Animation	☐
21(i)	Color dependence	☐
21(j)	Color & Contrast (OS)	☐
21(k)	Blinking objects	☐
21(l)	Forms	☐
22(a)	Text descriptions...	☐
22(b)	Synchronized Alternative	☐
22(j)	Flickering	☐
22(l)	Scripts	☐
22(p)	Time out	☐
24(c)	Captions	☐
24(d)	Video descriptions	☐
31(a)	Use without vision	☐
31(b)	Use with low vision	☐

Web-only test

Section 508 Baseline Test Report Checklist

Include this checklist when sharing results between agencies

Agency: _____ Agency contact details:_____

Product name: _____ Version # _____ Date tested: _____

Summary of main findings:

_____

_____

_____

#	Applicable Baseline requirement	Fail	Pass	N/A
1.	Keyboard navigation	☐	☐	
2.	Focus (visible)	☐	☐	
3.	Focus (order)	☐	☐	
4.	Focus (Revealing hidden content)	☐	☐	☐
5.	Skip-navigation links	☐	☐	☐
6.	Multi-state components	☐	☐	☐
7.	Images	☐	☐	☐
8.	Color (meaning)	☐	☐	☐
9.	Color (contrast)	☐	☐	
10.	Flashing (reserved)*	(☐)	(☐)	☐
11.	Forms (associated instructions)	☐	☐	☐
12.	Page Titles	☐	☐	
13.	Data Tables (headers)	☐	☐	☐
14.	Data Tables (cell-header mapping)	☐	☐	☐
15.	Headings	☐	☐	☐
16.	Links and user controls	☐	☐	☐
17.	Language	☐	☐	☐
16.	Audio (transcripts)	☐	☐	☐
19.	Video (descriptions)	☐	☐	☐
20.	Synchronized Media (captions)	☐	☐	☐
21.	Synchronized Media (descriptions)	☐	☐	☐
22.	Style-sheet non-dependence	☐	☐	☐
23.	Frames	☐	☐	☐
24.	Alternate pages	☐	☐	☐
25.	Time outs	☐	☐	☐
26.	Image maps	☐	☐	☐
27.	Plug-in Links	☐	☐	☐

* *Flashing must be tested, but there is no agreed-upon baseline test. Include evaluation methods in attached test report.*

#	Applicable 508 Std.	Fail
21(a)	Keyboard Accessibility	☐
21(c)	Visual Focus	☐
21(e)	Bitmap images	☐
21(k)	Blinking objects	☐
22(a)	Text descriptions...	☐
22(b)	Synchronized Alternative	☐
22(c)	Color dependence	☐
22(d)	Style Sheets	☐
22(e)	Server-side image maps	☐
22(f)	Client side (not server)	☐
22(g)	Row and column headers	☐
22(h)	Associate Data - Headers	☐
22(i)	Descriptive Frame Titles	☐
22(j)	Flickering	☐
22(k)	Alternative versions	☐
22(l)	Scripts	☐
22(m)	Plug-ins	☐
22(n)	Labels for forms	☐
22(o)	Skip Links	☐
22(p)	Time out	☐
24(c)	Captions	☐
24(d)	Video descriptions	☐
31(a)	Use without vision	☐
31(b)	Use with low vision	☐

Web+Software test

Section 508 Baseline Test Report Checklist

Include this checklist when sharing results between agencies

Agency: _____ Agency contact details:_____

Product name: _____ Version # _____ Date tested: _____

Summary of main findings:

_____

_____

_____

#	Baseline requirements	Fail	Pass	N/A
1.	Keyboard navigation	☐	☐	
2.	Focus (visible)	☐	☐	
3.	Focus (order)	☐	☐	
4.	Focus (Revealing hidden content)	☐	☐	☐
5.	Skip-navigation links	☐	☐	☐
6.	Multi-state components	☐	☐	☐
7.	Images	☐	☐	☐
8.	Color (meaning)	☐	☐	☐
9.	Color (contrast)	☐	☐	
10.	Flashing (reserved)*	(☐)	(☐)	☐
11.	Forms (associated instructions)	☐	☐	☐
12.	Page Titles	☐	☐	
13.	Data Tables (headers)	☐	☐	☐
14.	Data Tables (cell-header mapping)	☐	☐	☐
15.	Headings	☐	☐	☐
16.	Links and user controls	☐	☐	☐
17.	Language	☐	☐	☐
18.	Audio (transcripts)	☐	☐	☐
19.	Video (descriptions)	☐	☐	☐
20.	Synchronized Media (captions)	☐	☐	☐
21.	Synchronized Media (descriptions)	☐	☐	☐
22.	Style-sheet non-dependence	☐	☐	☐
23.	Frames	☐	☐	☐
24.	Alternate pages	☐	☐	☐
25.	Time outs	☐	☐	☐
26.	Image maps	☐	☐	☐
27.	Plug-in Links	☐	☐	☐
28.	Built-in accessibility features	☐	☐	☐

* *Flashing must be tested, but there is no agreed-upon baseline test. Include evaluation methods in attached test report.*

#	Section 508 Standard	Fail
21(a)	Keyboard Accessibility	☐
21(b)	Built-in Acc. Features	☐
21(c)	Visual Focus	☐
21(d)	Name, Role, State...	☐
21(e)	Bitmap images	☐
21(f)	Input text	☐
21(g)	Color & Contrast	☐
21(h)	Animation	☐
21(i)	Color dependence	☐
21(j)	Color & Contrast (OS)	☐
21(k)	Blinking objects	☐
21(l)	Forms	☐
22(a)	Text descriptions...	☐
22(b)	Synchronized Alternative	☐
22(c)	Color dependence	☐
22(d)	Style Sheets	☐
22(e)	Server-side image maps	☐
22(f)	Client side (not server)	☐
22(g)	Row and column headers	☐
22(h)	Associate Data - Headers	☐
22(i)	Descriptive Frame Titles	☐
22(j)	Flickering	☐
22(k)	Alternative versions	☐
22(l)	Scripts	☐
22(m)	Plug-ins	☐
22(n)	Labels for forms	☐
22(o)	Skip Links	☐
22(p)	Time out	☐
24(c)	Captions	☐
24(d)	Video descriptions	☐
31(a)	Use without vision	☐

#	Section 508 Standard	Fail
31(b)	Use with low vision	☐

Summary of failures

Attach this summary list to the Section 508 Baseline Test Report Checklist

Include, attach, or reference the location of the full test report.

Agency: _____ Agency contact details:_____

Product name: _____ Version # _____ Date tested: _____

#	Applicable Baseline	Baseline Failure (description and location)	Additional Notes

Document Content Change Log

Note: Minor punctuation, formatting and spelling changes not included.

Version 1.0.6, March 2015

First published version.

Version 1.1, February 2016

Location	Change
How the baseline tests are structured	Added “In Windows 8.1, testing is performed in Desktop mode.” Added Inspect URL for Windows 8.1. Added 8.1 SDK information to footnote.
Section 11, Forms	Added information about checking HTML version and ID naming in HTML5.
Section 13, Data Tables	Added information about use of rowgroup and colgroup. Added information about checking HTML version and deprecation of TD SCOPE in HTML5.
Section 14, Data Tables	Added information about use of rowgroup and colgroup. Added information about checking HTML version and deprecation of TD SCOPE in HTML5. Added information about ID naming in HTML 4.01 and HTML5.
Section 28, Built-in accessibility features	Added “system” to “Set the system text size to 200%”. Added Windows 8.1 instructions. Specified using Windows Key + U to open Ease of Access center. Changed instructions accordingly.

Version 2.0, October 2016

Location	Change
Agency issues beyond the test process	Removed “use of Heading <H> tags for style instead of structure, and using both an ALT and a TITLE attribute on an image where the two clearly contradict each other” as example of coding error. Added last bullet starting with “The baseline test methodology does not include guidance on managing a testing program. …”
Platforms, Browsers and Tools	Removed support for IE8, IE9. Added support for Windows 10. Added WAT 2015 to tools list. Added information on WAF and Color Contrast Analyser. Added content on the use of WAF, Chrome, and Firefox. Added “Notes on Browsers Differences” and “Notes on WAF”.
Various Sections	Added “Note: In Chrome, Inspect does not reveal the correct information for Flash content” and the note “Interactive Flash and embedded Java content should be tested in IE to determine the accessibility of the coded content”.
Section 5, Skip Links	Rationale: Changed title to “Repetitive Content” Changed “To enable equitable use by keyboard only users, there must be a method to skip past repetitive content. This can be provided by adding internal links to bypass repetitive content. Similarly, for screen reader users, if they must read content that is repeated on each page and cannot skip past it, their experience on the page can be very frustrating” to “To enable equitable use by keyboard only users, there must be a method to skip past repetitive content. Similarly, for screen reader users, if they must read content that is repeated on each page and cannot skip past it, their experience on the page

	can be very frustrating. A common method used to bypass repetitive content is internal (same page) links." Test Instruction 2: Added various instructions: location of skip target, testing if no skip links are marked, testing skip function, testing if no interactive target.
Section 7, Images	Rationale: Added "If font-based graphics are used to provide information, equivalent information must be provided in an accessible format." Test Instruction 1: Step a, added bullet: "Look for images that are rendered by using font-based graphics (e.g. up-arrows to indicate sort order, etc.)."
Section 8, Color (meaning)	Advisory: added bullet "WAF's Greyscale test works in more cases than WAT's, although there may be some sites where both work only partially or not at all."
Section 9, Color (Contrast)	Added "minimum" to "There must be contrasting colors/shades at a minimum ratio of 4.5:1 for discerning between background and foreground content."
Section 11, Forms (associated instructions)	Test Instruction 2: Added "Forms with ARIA labels should be tested in Chrome or Firefox to determine the accessibility of the coded forms."
Section 23, Frames	Advisory: Repeated content from "Notes on Using WAF" as relates to frames
Section 28, Built-in accessibility features	Rationale: Added Windows 10 Test Instruction 1b: Added Windows 10 Test Instruction 2b: Replaced "Open any text editing application (e.g., MS Word). Attempt to close an edited 2.0file without saving it first. The Sound Sentry indicator should flash" with instructions for Notepad, since previous test did not work reliably in different environments.
Throughout	Removed some references to page numbers and replaced them with references to section names

Version 2.0.1, November 2016

Location	Change
Various	Updated URLs from www.section508testing.org to www.dhs.gov/dhs-section-508-compliance-testing-tools