W3C

Web Content Accessibility Guidelines

W3C Working Draft 16-Mar-1999

This version:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316
Latest version:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH
Latest public version:
http://www.w3.org/TR/WD-WAI-PAGEAUTH
Previous version:
http://www.w3.org/TR/1999/WD-WAI-PAGEAUTH-19990226
Related Documents:
Techniques for Web Content Accessibility Guidelines
List of Checkpoints for the Web Content Accessibility Guidelines
Editors:
Wendy Chisholm <chisholm@trace.wisc.edu>
Gregg Vanderheiden <gv@trace.wisc.edu>
Ian Jacobs <ij@w3.org>

Copyright © 1999 W3C (MIT, INRIA, Keio), All Rights Reserved. W3C liability, trademark, document use and software licensing rules apply.

Abstract

This document provides guidelines to Web content developers [p. 24] (page authors and site designers) for making Web content [p. 24] accessible to people with disabilities. While these guidelines have been developed to help authors and authoring tools [p. 24] create accessible Web content, following them will also make Web content available to *all* users with different user agents [p. 26] (desktop browsers, voice browsers, mobile phones, automobile-based PCs, etc.) or operating under various constraints (noisy or noiseless surroundings, under- or over-illuminated rooms, in a hands-free environment, etc.). Following these guidelines will also help people find information on the Web more quickly. These guidelines do not discourage content developers from using images, video, etc., but rather explain how to make multimedia pages more accessible to a wide audience.

This document is part of a series of accessibility guidelines published by the W3C Web Accessibility Initiative. The series also includes User Agent Accessibility Guidelines ([WAI-USERAGENT] [p. 29]) and Authoring Tool Accessibility Guidelines ([WAI-AUTOOLS] [p. 28]).

Status of this document

This is a W3C Working Draft for review by the WAI Page Author Guidelines Working Group. It is published during the last call period for this document, which ends on March 19, 1999. This document incorporates many of the changes sent by reviewers as a result of last call. It is not entirely stable yet since not all of the suggestions have been incorporated. The editors will continue to incorporate comments and revise the document during and after last call.

This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". This is work in progress and does not imply endorsement by, or the consensus of, either W3C or members of the WAI GL Working Group.

This document has been produced as part of the W3C Web Accessibility Initiative, and is intended as a draft of a Proposed Recommendation for creating accessible Web content. The goal of the WAI Page Author Guidelines Working Group is discussed in the Working Group charter.

A list of current W3C Recommendations and other technical documents can be found at http://www.w3.org/TR.

Please send detailed comments on this document to w3c-wai-gl@w3.org.

Related documents

The current document includes an appendix entitled "List of Checkpoints for the Web Content Accessibility Guidelines." This appendix organizes all of the checkpoints [p. 7] defined in the current document by topic and priority [p. 7] . The checkpoints in the appendix link to their definitions in the current document. The topics identified in the appendix include images, multimedia, tables, frames, forms, and scripts.

A separate document, entitled "Techniques for Web Content Accessibility Guidelines," explains how to implement these checkpoints. It discusses each checkpoint in more detail and provides examples using HTML, Cascading Style Sheets (CSS), and Synchronized Multimedia Integration Language (SMIL) and MathML. **Note.** Not all browsers or multimedia tools may support the features described in the guidelines. In particular, new features of HTML 4.0 or CSS 1 or CSS 2 may not be supported. To help readers find out about support, the Techniques Document includes:

- a section on browser support
- an index of HTML elements and attributes.
- a checkpoint map that shows where each checkpoint is discussed in the Techniques Document.

The Techniques Document will be updated more regularly than the current document in order to track changes in technology.

Available formats

This document is available in the following formats:

HTML:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.html
A plain text file:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.txt,
HTML (Guidelines, List of Checkpoints, Techniques) as a gzip'ed tar file :
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.tgz,
HTML (Guidelines, List of Checkpoints, Techniques) as a zip archive:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.zip,
A PostScript file:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.ps,
A PDF file:
http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316/wai-pageauth.pdf.

The definitive version of this document is the HTML version.

Table of Contents

Abstract 1
Status of this document 2
1. Introduction 5
2. Themes of Accessible Design 6
    1. Ensuring Graceful Transformation 6
    2. Making Content Understandable and Navigable 6
3. How the Guidelines are Organized 7
4. Priorities 7
5. Conformance 7
6. Web Content Accessibility Guidelines 9
    1. Provide text equivalents for visual information. 9
    2. Provide descriptions of visual information. 10
    3. Provide text equivalents for audio information. 11
    4. Don’t rely on color alone. 11
    5. Use markup and style sheets properly. 12
    6. Clarify natural language usage 13
    7. Create tables that transform gracefully. 14
    8. Ensure that pages featuring new technologies transform gracefully. 15
    9. Ensure user control of time-sensitive content changes. . . . 16
    10. Ensure direct accessibility of embedded user interfaces. . . . 17
    11. Design for device-independence. 17
    12. Use interim solutions. 18
    13. Use W3C technologies and guidelines. 19
    14. Provide context and orientation information. 20
    15. Provide clear navigation mechanisms. 21
    16. Ensure that documents are clear and simple. 22
Appendix A. - Validation 22
Appendix B. - Definitions 23
Acknowledgments 26
References 28

Additional appendix: List of Checkpoints for the Web Content Accessibility Guidelines.

1. Introduction

For those unfamiliar with accessibility issues pertaining to Web page design, consider that many users may be operating in contexts very different from your own:

- They may not be able to see, hear, or move. may not be able to process some types of information easily or at all.
- They may have difficulty reading or comprehending text.
- They may not have or be able to use a keyboard or mouse.
- They may have a text-only screen, a small screen, or a slow Internet connection.
- They may not speak or understand fluently the language in which the document is written.
- They may be in a situation where their eyes, ears, or hands are busy or interfered with (e.g., driving to work, working in a loud environment, etc.).
- They may have an early version of a browser, a different browser entirely, a voice browser, or a different operating system.

Content developers must consider these different situations during page design. While there are several situations to consider, each accessible design choice generally benefits several disability groups at once and the Web community as a whole. For example, by using style sheets [p. 26] to control font styles and eliminating the FONT element, HTML authors will have more control over their pages, make those pages more accessible to people with low vision, and by sharing the style sheets, will often shorten page download times for all users.

The guidelines discuss accessibility issues and provide accessible design solutions. They enumerate typical scenarios (similar to the font style example) that may pose problems for users with certain disabilities. For example, the first guideline [p. 9] explains how content developers can make images accessible. Some users may not be able to see images, others may use text-based browsers that do not support images, while others may have turned off support for images (e.g., due to a slow Internet connection). The guidelines do not suggest avoiding images as a way to improve accessibility. Instead, they explain that providing a text equivalent [p. 25] that states the purpose of the image will make it accessible.

How does a text equivalent make the image accessible? Users with blindness or low vision can understand the function of the image when the text is read aloud by a speech synthesizer. This is particularly important when the image is part of a hyperlink since, without an explanation of the link's destination, a user with blindness wouldn't know whether to follow it. In addition to benefitting users with disabilities, text equivalents can be used by search robots when indexing your pages.

2. Themes of Accessible Design

The guidelines address two general themes: ensuring graceful transformation, and making content understandable and navigable.

1. Ensuring Graceful Transformation

By following these guidelines, content developers can create pages that transform gracefully. Pages that transform gracefully remain accessible despite any of the constraints described in the introduction [p. 5] , including physical, sensory, and cognitive disabilities, work constraints, and technological barriers. Here are some keys to designing pages that transform gracefully:

- Separate content from structure from presentation. The appendix explains the difference between content, structure, and presentation [p. 24]).
- Provide text (including text equivalents, [p. 25] and text descriptions [p. 25]). Textual information can be rendered in ways that are available to almost all browsing devices and accessible to almost all users.
- Create documents that only work even if the user cannot see and/or hear. Some content will be sensory-specific (e.g., audio, video, applets that present visual information), so provide equivalent information in forms suited to other senses as well. **Note.** This does not mean creating an entire auditory version of a site. Screen readers [p. 26] will be able to speak all information on a page as long as it is available in text.
- Create documents that do not rely on one type of hardware. Pages should be usable by people without mice, with small screens, low resolution screens, black and white screens, no screens, with only voice or text output, etc.

The theme of graceful transformation is addressed primarily by guidelines 1 to 13.

2. Making Content Understandable and Navigable

Content developers should make content understandable and navigable. This includes not only making the language clear and simple, but also providing understandable features for navigating within and between pages. Providing navigation tools and orientation information in pages will maximize accessibility and usability. Remember, not all users can make use of visual clues such as image maps, proportional scroll bars, side-by-side frames, or graphics that guide sighted users of graphical desktop browsers. Users also lose contextual information when they can only view a portion of a page, either because they are accessing the page one word at a time (speech synthesis or braille display [p. 24]), or one section at a time (small display, or a magnified display). Very large tables, lists, menus, etc. without orientation information may be very disorienting to users.

The theme of making content understandable and navigable is addressed primarily in guidelines 14 to 16.

3. How the Guidelines are Organized

This document includes sixteen guidelines, or general principles of accessible design. Each guideline is followed by a more detailed statement of the principle and the rationale behind the guideline.

The most important part of each guideline is the list of checkpoints explaining how the guideline applies in typical content development scenarios. Each checkpoint definition links to a section of the Techniques Document where implementations and examples of the checkpoint are discussed.

Each checkpoint is specific enough so that someone reviewing a page or site may verify that the checkpoint has been satisfied.

4. Priorities

Each checkpoint has priority level assigned by the Working Group based on the checkpoint's impact on accessibility.

[Priority 1] or [Priority One]
: A Web content developer **must** satisfy this checkpoint. Otherwise, one or more groups will find it impossible to access information in the document. Satisfying this checkpoint is a basic requirement for some groups to be able to use Web documents.

[Priority 2] or [Priority Two]
: A Web content developer **should** satisfy this checkpoint. Otherwise, one or more groups will find it difficult to access information in the document. Satisfying this checkpoint will remove significant barriers to accessing Web documents.

[Priority 3] or [Priority Three]
: A Web content developer **may** address this checkpoint. Otherwise, one or more groups will find it somewhat difficult to access information in the document. Satisfying this checkpoint will improve access to Web documents.

Some checkpoints specify a priority level that may change under certain (indicated) conditions.

5. Conformance

This document defines three conformance levels:

- **Conformance Level A**: A document or process satisfies all Priority 1 checkpoints, or
- **Conformance Level Double-A**: A document or process satisfies all Priority 1 and 2 checkpoints, or
- **Conformance Level Triple-A**: A document or process satisfies all Priority 1, 2, and 3 checkpoints.

Claims of conformance to this document must provide the following information:

1. Which conformance level has been achieved: (Conformance Level A, Double-A, or Triple-A)
2. The title of this Guidelines document: "Web Content Accessibility Guidelines"
3. The URI of this Guidelines document, which must be written like this: http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316

Following is an example of a conformance claim:

The page available at http://www.something.com/document/cover.html meets Conformance Level Double-A as defined in "Web Content Accessibility Guidelines", located at http://www.w3.org/WAI/GL/WD-WAI-PAGEAUTH-19990316.

6. Web Content Accessibility Guidelines

Guideline 1. Provide text equivalents for visual information.

Provide text equivalents for all images, applets, and image maps.

A text equivalent [p. 25] describes the purpose or function of an image, applet, image map, or other information.

If text equivalents are not provided for visual information, people who are blind, have low vision, or any user who cannot or has chosen not to view graphics will not know the purpose of the visual components on the page.

The Techniques Document explains how to write text equivalents specifically for images used as image maps, as spacers, as bullets in lists, as graphical buttons, as links, or to present mathematical equations.

Checkpoints: 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8

1.1 Provide text equivalents for all images [Priority 1]
For example, in HTML, specify a text equivalent [p. 25] via the "alt" attribute of the IMG and INPUT elements, or for OBJECT, via the element's content.
Techniques for checkpoint 1.1

1.2 Provide text equivalents for all applets and other programmatic objects. [Priority 1]
For example, in HTML, provide a text equivalent for OBJECT via the element's content. Also in HTML, for APPLET, specify it via the "alt" attribute or within the content of the element.
Refer also to checkpoint 8.5 and guideline 10.
Techniques for checkpoint 1.2

1.3 For short animations such as "animated gifs," provide a text equivalent and a description [p. 25] if needed. [Priority 1]
Refer also to guideline 2.
Techniques for checkpoint 1.3

1.4 Provide a text equivalent for each active region of an image map. [Priority 1]
For example, in HTML, provide information via the "alt" attribute of the AREA element or the MAP element with A elements as content.
Techniques for checkpoint 1.4

1.5 Provide client-side image maps instead of server-side image maps except where the regions cannot be defined with an available geometric shape. [Priority 1]
Techniques for checkpoint 1.5

1.6 Replace ASCII art with an image or describe the ASCII art and provide a means (e.g., a link) to skip over it. [Priority 1 or Priority 2 depending on the importance of the information.]
Note. When replacing ASCII art with an image, specify a text equivalent [p. 25] for the image. The guidelines includes an example of ascii art [p. 23] .
Techniques for checkpoint 1.6

1.7 Provide redundant text links for each active region of an image map. [Priority 1 - if server-side image maps [p. 25] are used, Priority 2 - if client-side image maps [p. 25] are used. Content developers will not need to provide redundant text links for client-side image maps once most user agents render text equivalents for the map links.]

Note. If the description of important ASCII art is long, provide a description in addition to a text equivalent. Refer also to guideline 2.

Techniques for checkpoint 1.7

1.8 Provide individual button controls in a form rather than simulating a set of buttons with an image map. [Priority 2]

Techniques for checkpoint 1.8

Note. When a button has an image, specify text equivalent [p. 25] for the image.

Guideline 2. Provide descriptions of visual information.

Provide descriptions of information in graphics, scripts, applets, videos, or animations if the information is not fully described through text equivalents or by the primary text.

A description [p. 25] provides information about the appearance or sound of an image, script, or applet. Descriptions make information presented graphically (charts, billboards, diagrams) perceivable to people with blindness, some people with low vision, and to people who have chosen not to view graphics, scripts, or applets or whose browser does not support scripts or applets.

In movies or visual presentations, visual action such as body language or other visual cues may not be accompanied by enough audio information to convey the same information. Unless verbal descriptions of this visual information are provided, people who cannot see (or look at) the visual content will not be able to perceive it.

If a visual presentation has an associated auditory presentation (e.g., a movie), synchronize the audio version of the descriptions with the existing auditory presentation, and collate [p. 24] a text version of the descriptions with the text transcripts (captions) of the primary audio track. The collated information will make the presentation available to people who are deaf-blind and to people who cannot play or choose not to play movies, animations, etc.

Checkpoints: 2.1, 2.2

2.1 Provide a description of each graphic, script, or applet that conveys important [p. 25] information. [Priority 1]

For example, in HTML, for IMG specify a description via "longdesc" and for OBJECT within the element’s content. Or, put a description link [p. 25] in the document.

Techniques for checkpoint 2.1

2.2 For movies, provide auditory descriptions that are synchronized with the original audio. [Priority 1]
Techniques for checkpoint 2.2

Guideline 3. Provide text equivalents for audio information.

Provide text transcripts, text descriptions, or captions of auditory events that occur in audio and video.

When a transcript is synchronized with a video presentation it is called a "caption". Captions are used by people who cannot hear the audio track of the video material. Without transcripts and captions, people who are deaf, or hard of hearing, or any user who cannot or has chosen not to hear sound cannot perceive the information presented through speech, sound effects, music, etc.

Checkpoints: 3.1, 3.2, 3.3, 3.4

3.1 For stand-alone audio files, provide a text transcript of all words (spoken or sung) and all important sounds. [Priority 1]
Techniques for checkpoint 3.1
3.2 For audio associated with video, synchronize the text transcript with the video. [Priority 1]
Synchronize the transcript (of dialog and sounds) as a caption.
Techniques for checkpoint 3.2
3.3 Where sounds are played automatically, provide visual notification and transcripts. [Priority 1 or Priority 2 depending on the importance of the sound.]
Techniques for checkpoint 3.3
3.4 Provide a text version of the auditory description and collate it with the text transcript (captions) of the primary audio track. [Priority 2]
Techniques for checkpoint 3.4

Guideline 4. Don’t rely on color alone.

Ensure that text and graphics are understandable when viewed without color.

If color alone is used to convey information, people who cannot differentiate between certain colors and users with devices that have non-color or non-visual displays will not receive the information. When foreground and background colors are too close to the same hue, they may not provide sufficient contrast when viewed using monochrome displays or by people with different types of color deficits.

Checkpoints: 4.1, 4.2

4.1 Ensure that all information conveyed with color is also available without color, for example from context or markup. [Priority 1]

When text has a *purely* decorative value and conveys no information other than the color itself, it is not necessary to provide that information elsewhere.

Techniques for checkpoint 4.1

4.2 Ensure that foreground and background color combinations provide sufficient contrast when viewed by someone having color deficits or when viewed on a black and white screen. [Priority 2]

Techniques for checkpoint 4.2

Guideline 5. Use markup and style sheets properly.

Mark up documents with the proper structural elements. Control presentation with style sheets rather than with presentation elements and attributes.

Using markup improperly -- not according to specification -- hinders accessibility. Misusing markup for a presentation effect (e.g., using a table for layout or a header to change the font size) makes it difficult for users with specialized software to understand the organization of the page or to navigate through it. Furthermore, presentation effects used alone to convey (e.g., constructing what looks like a table of data with an HTML PRE element) make it difficult to render a page intelligibly to other devices (refer to the description of difference between content, structure, and presentation [p. 24]).

Content developers may be tempted to use (or misuse) constructs that achieve a desired formatting effect on older browsers. They must be aware that these practices cause accessibility problems and must consider whether the formatting effect is so critical as to make the document inaccessible to some users.

At the other extreme, content developers must not sacrifice *correct* markup because they are concerned about the accessibility problems it might cause. For example, it is appropriate to use the TABLE element in HTML to mark up tabular data. Doing so (and creating tables that transform gracefully (refer to guideline 7) make it possible for software to render tables other than as two-dimensional grids.

Checkpoints: 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7

5.1 Use header elements to convey logical structure and use them according to specification. [Priority 2]

For example, in HTML, use H2 to indicate a subsection of H1. Do not use headers for font effects.

Techniques for checkpoint 5.1

5.2 Mark up lists and list items properly. [Priority 2]

For example, in HTML, nest OL, UL, and DL lists properly.

Techniques for checkpoint 5.2

5.3 Mark up quotations. Do not use quotation markup for formatting effects such as indentation. [Priority 2]

For example, in HTML, use the Q and BLOCKQUOTE elements to markup short and longer quotations, respectively.

Techniques for checkpoint 5.3

5.4 Create documents that validate to published formal grammars. [Priority 2]

For example, include a document type declaration at the beginning of a document that refers to a published DTD.

Techniques for checkpoint 5.4

5.5 Use style sheets to control layout and presentation. [Priority 2]

For example, use the CSS 'font' property instead of the HTML FONT element to control font styles.

Techniques for checkpoint 5.5

5.6 When an appropriate markup language exists, use markup rather than images to convey information. [Priority 2]

For example, use MathML to mark up mathematical equations, and style sheets [p. 26] to format text and control layout. Also, avoid using images to represent text - use text and style sheets instead.

Note. If important information is conveyed in many images on the page, provide an alternative accessible page. [p. 20] Refer also to guideline 8 and guideline 13.

Techniques for checkpoint 5.6

5.7 Use relative rather than absolute units in markup language attribute values and style sheet property values. [Priority 2]

For example, in CSS, use 'em' or percentage lengths rather than 'pt' or 'cm', which are absolute units.

Techniques for checkpoint 5.7

Guideline 6. Clarify natural language usage

Use markup that facilitates pronunciation or interpretation of abbreviated or foreign text.

Changes between multiple languages on the same page and abbreviations can both be indecipherable when machine-spoken or brailled unless they are identified. Content developers should identify the predominant natural language of a document's text and indicate when language changes occur. They should also provide expansions of abbreviations and acronyms. This information also helps search engines find key words and identify documents in a desired language. The information also improves readability of the Web for all people, including those with learning disabilities, cognitive disabilities, and deafness.

Checkpoints: 6.1, 6.2, 6.3

6.1 Clearly identify changes in the natural language of a document's text. [Priority 1]
For example, in HTML use the "lang" attribute. In XML, use "xml:lang". Server operators should configure their server to take advantage of the content negotiation mechanisms of the HTTP protocol so that clients can automatically retrieve documents of the preferred language.
Techniques for checkpoint 6.1
6.2 Specify the expansion of abbreviations and acronyms. [Priority 2 for the first occurrence of the acronym or abbreviation in a given document, Priority 3 thereafter.]
For example, in HTML, use the "title" attribute of the ABBR and ACRONYM elements.
Techniques for checkpoint 6.2
6.3 Identify the primary natural language of a document. [Priority 3]
For example, in HTML, set the "lang" attribute on the HTML element. In XML, use "xml:lang".
Techniques for checkpoint 6.3

Guideline 7. Create tables that transform gracefully.

Ensure that tables have necessary markup to be transformed by accessible browsers and other user agents.

Tables should be used to mark up truly tabular data ("data tables"). Content developers should avoid using them to lay out pages ("layout tables"). Tables for any use also present special problems to users of screen readers [p. 26] .

Many user agents [p. 26] transform tables to present them and if not marked up properly, the tables will not make sense when rendered. Refer also to guideline 5.

The following checkpoints will directly benefit people who access a table through auditory means (e.g., a screen reader or an Automobile PC that operates by speech input and output) or who view only a portion of the page at a time (e.g., users with blindness or low vision using speech output or a braille display, [p. 24] or other users of devices with small displays, etc.).

Checkpoints: 7.1, 7.2, 7.3, 7.4, 7.5, 7.6

7.1 For data tables, identify headers for rows and columns. [Priority 1]
For example, in HTML, use TD to identify data cells and TH to identify headers.
Techniques for checkpoint 7.1
7.2 For data tables that have more than one row and/or more than one column of header cells, use markup to associate data cells and header cells. [Priority 1]
For example, in HTML, use THEAD, TFOOT, and TBODY to group rows, COL and COLGROUP to group columns, and the "axis", "scope", and "headers" attributes, to describe more complex relationships among data.
Techniques for checkpoint 7.2

7.3 Avoid using tables for layout. [Priority 2]
Techniques for checkpoint 7.3
7.4 If a table is used for layout, do not use any structural markup for the purpose of visual formatting. [Priority 2]
For example, in HTML do not use the TH element to cause the content of a (non-table header) cell to be displayed centered and in bold. Use the "summary" attribute to explain that the table is a layout table.
Techniques for checkpoint 7.4
7.5 Provide summaries for tables. [Priority 3]
For example, in HTML, use the "summary" attribute of the TABLE element.
Techniques for checkpoint 7.5
7.6 Provide abbreviations for header labels. [Priority 3]
For example, in HTML, use the "abbr" attribute on the TH element.
Techniques for checkpoint 7.6

Refer also to checkpoint 12.3 and checkpoint 5.6.

Guideline 8. Ensure that pages featuring new technologies transform gracefully.

Ensure that pages are accessible even when newer technologies are not supported or are turned off.

Although content developers are encouraged to use new technologies that solve problems raised by existing technologies, they should know how to make their pages still work with older browsers and people who choose to turn off features.

Checkpoints: 8.1, 8.2, 8.3, 8.4, 8.5

8.1 Ensure that descriptions and text alternatives for dynamic content are updated when the dynamic content changes. [Priority 1]
Techniques for checkpoint 8.1
8.2 For scripts that present important [p. 25] information or functionality, provide an equivalent. (Refer to the definition of equivalent. [p. 25]) [Priority 1]
For example, in HTML use NOSCRIPT or a server-side script that outputs the equivalent.
Techniques for checkpoint 8.2
8.3 For pages that use style sheets or presentation markup [p. 26] , ensure that the content of each page is organized logically. [Priority 1]
This makes it more likely that the document will be understood even when styles are turned off or overridden by the user. For applets and programmatic objects, provide text equivalents [p. 25] and descriptions. [p. 25]
Techniques for checkpoint 8.3
8.4 Provide an alternative presentation or page when the primary content is dynamic. [Priority 2]
For example: In HTML, use NOFRAMES at the end of each frameset, NOSCRIPT for every script, and server-side instead of client-side scripts.

Techniques for checkpoint 8.4

8.5 Provide non-text equivalents for all applets and other programmatic objects. [Priority 2]

For example, for an animation, one might provide a snapshot of the animation. For an applet, one might provide a video equivalent. This checkpoint is helpful for people whose browsers cannot run the applets or programmatic objects but who can perceive visual presentations. It can also be helpful to non-readers, including deaf non-readers. For example, for an applet, one might provide a video equivalent of a person communicating the content using a manual communication system (e.g., sign language).

Refer also to checkpoint 1.2.

Techniques for checkpoint 8.5

Refer also to checkpoint 13.4.

Guideline 9. Ensure user control of time-sensitive content changes.

Ensure that moving, blinking, scrolling, or auto-updating objects or pages may be paused or stopped.

Some people with cognitive or visual disabilities are unable to read moving text quickly enough or at all. Movement can also cause such a distraction that the rest of the page becomes unreadable for people with cognitive disabilities. Screen readers [p. 26] are unable to read moving text. People with physical disabilities might not be able to move quickly or accurately enough to interact with moving objects. People with photosensitive epilepsy can have seizures triggered by flickering or flashing in the 4 to 59 flashes per second (Hertz) range with a peak sensitivity at 20 flashes per second as well as quick changes from dark to light (like strobe lights).

Checkpoints: 9.1, 9.2, 9.3, 9.4

9.1 Until user agents provide the ability to stop the refresh, do not create periodically auto-refreshing pages [Priority 1]

For example, in HTML, don't cause pages to auto-refresh with "HTTP-EQUIV=refresh" until user agents allow users to turn off that feature.

Techniques for checkpoint 9.1

9.2 Avoid any blinking or updating of the screen that causes flicker. [Priority 1]

Techniques for checkpoint 9.2

9.3 Avoid movement in pages. When a page includes moving content, provide a mechanism to allow users to freeze motion or updates in applets and scripts or use style sheets and scripting to create movement. [Priority 2]

Note. Style sheets and scripts can be turned off or overridden.

Refer also to guideline 10.

Techniques for checkpoint 9.3

9.4 Until user agents provide the ability to stop auto-redirect, do not use markup to redirect pages automatically. Instead, configure the server to perform redirects. [Priority 2]

Techniques for checkpoint 9.4

Note. The BLINK and MARQUEE elements are not defined in any W3C HTML specification and should not be used. Refer also to guideline 13.

Guideline 10. Ensure direct accessibility of embedded user interfaces.

Ensure that the user interface follows principles of accessible design: device-independent access to functionality, keyboard operability, self-voicing, etc.

When an embedded object has its "own interface", the interface -- like the interface to the browser itself -- must be accessible. If the interface of the embedded object cannot be made accessible, an alternative accessible solution must be provided.

Note. For information about accessible interfaces, please consult the User Agent Accessibility Guidelines ([WAI-USERAGENT] [p. 29]), the Authoring Tool Accessibility Guidelines ([WAI-AUTOOL] [p. 29]), and the discussion of applets and other objects in the Techniques Document.

Checkpoint: 10.1

10.1 Make programmatic elements such as scripts and applets directly accessible or compatible with assistive technologies [Priority 1 if functionality is important [p. 25] and not presented elsewhere, otherwise Priority 2.]

Refer also to guideline 8.

Techniques for checkpoint 10.1

Guideline 11. Design for device-independence.

Use features that enable activation of page elements via input devices other than a pointing device (e.g., keyboard, voice, etc.).

Interaction with a document must not depend on a particular input device; remember not everyone uses a mouse. If, for example, a form control can only be activated with a mouse or other pointing device, someone who is using the page without sight, with voice input, or with a keyboard or who is using an input device other than a mouse (e.g., a braille display [p. 24]) will not be able to use the form.

Note. Providing text equivalents for image maps or images used as links makes it possible for users to interact with them without a pointing device. Refer also to guideline 1.

Generally, pages that allow keyboard interaction are also accessible through speech input or a command line interface.

Checkpoints: 11.1, 11.2, 11.3, 11.4

11.1 Ensure that all elements that have their own interface are keyboard operable. [Priority 2]
Refer also to guideline 10.
Techniques for checkpoint 11.1
11.2 For scripts, specify logical event handlers rather than device-dependent event handlers. [Priority 2]
For example, in HTML use "onfocus", "onblur", and "onselect".
Techniques for checkpoint 11.2
11.3 Create a logical tab order through links, form controls, and objects. [Priority 3]
For example, in HTML, specify tab order via the "tabindex" attribute or ensure a logical page design.
Techniques for checkpoint 11.3
11.4 Provide keyboard shortcuts to links, including those in client-side image maps, [p. 25] form controls, and groups of form controls. [Priority 3]
For example, in HTML, specify shortcuts via the "accesskey" attribute.
Techniques for checkpoint 11.4

Guideline 12. Use interim solutions.

Use interim accessibility solutions so that assistive technologies and older browsers will operate correctly.

For example, older browsers do not allow users to navigate to empty edit boxes. Older screen readers read lists of consecutive links as one link. These active elements are therefore difficult or impossible to access. Also, changing the current window or popping up new windows can be very disorienting to users who have available, but aren't using, the graphical features of the desktop environment.

Note. The following checkpoints apply until most users are able to secure newer technologies that address these issues.

Checkpoints: 12.1, 12.2, 12.3, 12.4, 12.5

12.1 Do not cause pop-ups or other windows to appear and do not change the current window without informing the user. [Priority 2]
For example, in HTML, avoid using a frame whose target is a new window.
Techniques for checkpoint 12.1
12.2 For all form controls with implicitly associated labels, ensure that the label is properly positioned. [Priority 2]
The label must immediately precede its control on the same line (allowing more than one control/label per line) or be in the line preceding the control (with only one label and one control per line).
Techniques for checkpoint 12.2

12.3 Provide a linear text alternative (on the current page or some other) for *all* tables that lay out text in parallel, word-wrapped columns. [Priority 2]

Note. This checkpoint benefits people with user agents [p. 26] (such as some screen readers [p. 26]) that are unable to handle blocks of text presented side-by-side.

Techniques for checkpoint 12.3

12.4 Include default, place-holding characters in edit boxes and text areas. [Priority 3]

For example, in HTML, do this for TEXTAREA and INPUT.

Techniques for checkpoint 12.4

12.5 Include non-link, printable characters (surrounded by spaces) between links that occur consecutively. [Priority 3]

Techniques for checkpoint 12.5

Guideline 13. Use W3C technologies and guidelines.

Use W3C technologies (according to specification) and follow accessibility guidelines. Where it is not possible to use a W3C technology, or doing so results in material that does not transform gracefully, provide an alternative version of the content that is accessible.

Many non-W3C technologies (e.g., PDF, Shockwave, and other data formats) used to encode information require either plug-ins or stand-alone applications which often create pages that cannot be viewed or navigated using standard Web access or screen reading tools. Avoiding non-W3C and non-standard features (proprietary elements, attributes, properties, and extensions) will tend to make pages more accessible to more people using a wider variety of hardware and software.

Even when W3C technologies are used, they must be used in accordance with accessibility guidelines.

Note. Converting documents (from PDF, PostScript, RTF, etc.) to W3C markup languages (HTML, XML) does not always create an accessible document. Therefore, test each page for accessibility and usability after the conversion process. If a page does not readily convert, either revise the page until its original representation converts appropriately or provide an HTML or plain text equivalent.

Checkpoints: 13.1, 13.2, 13.3, 13.4

13.1 If W3C technologies are used, use the latest W3C specification where it is supported. [Priority 2]

Techniques for checkpoint 13.1

13.2 If W3C technologies are used, avoid deprecated language features. [Priority 2]

Techniques for checkpoint 13.2

13.3 Provide information so that users may receive documents according to their preferences (e.g., language, content type, etc.) [Priority 3]

Note. Use content negotiation where possible.

Techniques for checkpoint 13.3

13.4 If, after best efforts [p. 20] , you cannot avoid using a non-W3C technology or any W3C technology in an accessible way, provide a link to an alternative page that uses W3C technologies, is accessible, has equivalent information, and is updated as often as the inaccessible (original) page. [Priority 1]

Techniques for checkpoint 13.4

Note. Content developers should only resort to alternative pages when other solutions fail because alternative pages are generally updated less often than "primary" pages. An out-of-date page may be as frustrating as one that is inaccessible since, in both cases, the information presented on the original page is not available. Automatically generating alternative pages may lead to more frequent updates, but content developers must still be careful to ensure that generated pages always make sense and that users are able to navigate a site by following links on primary pages, alternative pages, or both. Before resorting to an alternative page, reconsider the design of the original page; simplifying it is likely to make it more effective for all users.

Guideline 14. Provide context and orientation information.

Provide context and orientation information to help users understand complex pages or elements

Grouping and providing contextual information about the relationships between elements can be useful for all users. Complex relationships between elements on a page may be difficult for people with cognitive disabilities and people with visual disabilities to interpret.

Checkpoints: 14.1, 14.2, 14.3, 14.4, 14.5

14.1 Title each frame so that users can keep track of frames by title. [Priority 1]

For example, in HTML use the "title" attribute on FRAME elements.

Techniques for checkpoint 14.1

14.2 Describe the purpose of frames and how frames relate to each other if it is not obvious by frame titles alone. [Priority 2]

For example, in HTML, use "longdesc," or a description link [p. 25]

Techniques for checkpoint 14.2

14.3 Group form controls. [Priority 2 - for radio buttons and checkboxes, Priority 3 - for other controls.]

For example, in HTML use the FIELDSET and LEGEND elements.

Techniques for checkpoint 14.3

14.4 Associate labels explicitly with their controls. [Priority 2]

For example, in HTML use LABEL and its "for" attribute.

Techniques for checkpoint 14.4

14.5 Divide long lists of choices into manageable groups. [Priority 2]
For example, in HTML use the OPTGROUP element.
Techniques for checkpoint 14.5

Guideline 15. Provide clear navigation mechanisms.

Provide clear and consistent navigation mechanisms - orientation information, navigation bars, a site map, etc. - to increase the likelihood that a person will find what they are looking for at a site.

Clear and consistent navigation mechanisms [p. 26] will benefit not only people with cognitive disabilities or blindness, but everyone who visits the site.

Checkpoints: 15.1, 15.2, 15.3, 15.4, 15.5, 15.6, 15.7, 15.8, 15.9, 15.10

15.1 Make link phrases terse yet meaningful when read on their own or in succession. [Priority 2]
Avoid general phrases, such as "click here" (which is device-dependent in addition to saying nothing about what is to be found at the end of the link).
Techniques for checkpoint 15.1
15.2 Provide metadata to add semantic information to pages and sites. [Priority 2]
For example, use RDF ([RDF] [p. 28]) to indicate the document's author, the type of content, and to describe the navigation mechanism, etc.
Techniques for checkpoint 15.2
15.3 Provide navigation bars to highlight and give access to the navigation mechanism. [Priority 3]
Note. Some user agents [p. 26] can build navigation tools from document relations described by the HTML LINK element and "rel" or "rev" attributes (e.g., rel="next", rel="previous", rel="index", etc.)
Techniques for checkpoint 15.3
15.4 Provide a site map or table of contents that makes the structure of a Web site apparent and facilitates navigation. [Priority 3]
Techniques for checkpoint 15.4
15.5 Provide a description of the general layout of the site, the access features provided, and how to use them. [Priority 3]
Techniques for checkpoint 15.5
15.6 Use navigation mechanisms in a consistent manner. [Priority 3]
Techniques for checkpoint 15.6
15.7 Enable different types of searches for different skill levels and preferences. [Priority 3]
Techniques for checkpoint 15.7
15.8 Place distinguishing information at the beginning of headings, paragraphs, lists, etc. [Priority 3]
Note. This is commonly referred to as "front-loading" and is especially helpful for people accessing information with serial devices such as speech synthesizers.
Techniques for checkpoint 15.8

15.9 Facilitate off-line browsing by creating a single downloadable file for documents that exist as a series of separate pages. [Priority 3]

A second option is to provide an archive (e.g., with zip, gzip, stuffit, etc.) of the multiple pages.

Techniques for checkpoint 15.9

15.10 Group related links, identify the group (for user agents), and, until user agents do so, provide a way to bypass the group. [Priority 3]

For example, when creating a navigation bar composed of links in HTML use "title" on FRAME, DIV, SPAN, etc. Use class="nav" to identify the group. Use "tabindex=1" on an anchor after the group so users may quickly navigate to it.

Techniques for checkpoint 15.10

Guideline 16. Ensure that documents are clear and simple.

Ensure that documents are clear and simple to promote comprehension.

Consistent page layout, recognizable graphics, and easy to understand language benefit all who visit a site. In particular, they help people with cognitive disabilities or who have difficulty reading. (However, ensure that images have text equivalents for people who are blind, have low vision, or for any user who cannot or has chosen not to view graphics. Refer also to guideline 1.)

Using clear and simple language promotes effective communication. Conditions such as cognitive disabilities, learning disabilities, and deafness can make access to written information difficult to impossible for some users. This consideration also applies to the many people whose first language differs from your own.

Checkpoints: 16.1, 16.2, 16.3

16.1 Use language that is clear and simple, yet appropriate for the site's content. [Priority 1]

Techniques for checkpoint 16.1

16.2 Provide icons or graphics (with text equivalents and descriptions) where they facilitate comprehension of the page. [Priority 3]

Note. These are particularly important for non-readers. Refer also to 1 [p. 9] and 2 [p. 10]

Techniques for checkpoint 16.2

16.3 Create a style of presentation that is consistent across pages. [Priority 3]

Techniques for checkpoint 16.3

Appendix A. - Validation

Validate accessibility with automatic tools and human review. Automated methods are generally rapid and convenient but cannot identify all accessibility issues. Human review can help ensure clarity of language and

ease of navigation.

Begin using validation methods at the earliest stages of development. Accessibility issues identified early are easier to correct and avoid.

Following are some important validation methods, discussed in more detail in the section on validation in the Techniques Document.

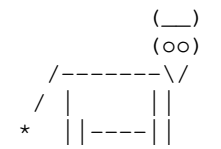
1. Use an automated accessibility tool and browser validation tool. Please not that software tools do not address all accessibility issues, such as the meaningfulness of link text, the applicability of a description [p. 25] , etc.
2. Validate the HTML.
3. Validate the style sheets.
4. Use a text-only browser or emulator.
5. Use multiple graphic browsers, with:
   - sounds and graphics loaded,
   - graphics not loaded,
   - sounds not loaded,
   - no mouse,
   - frames, scripts, style sheets, and applets not loaded
6. Use several browsers, old and new.
7. Use a self-voicing browser, a screen reader, magnification software, a small display, etc.
8. Use spell and grammar checkers. A person reading a page with a speech synthesizer may not be able to decipher the synthesizer's best guess for a word with a spelling error. Eliminating grammar problems increases comprehension.
9. Review content and structure [p. 24] for clarity and simplicity. Readability statistics, such as those generated by some word processors may be useful indicators of clarity and simplicity. Better still, ask an experienced (human) editor to review written content for clarity. Editors can also improve the usability of documents by identifying intercultural problems that might arise due to language or icon usage.
10. Invite people with disabilities to review your documents. Expert and novice users with disabilities will provide valuable feedback about accessibility or usability problems and their severity.

Appendix B. - Definitions

Accessible
: ???

Applet
: A program inserted into a Web page.

ASCII art
: ASCII art refers to text characters and symbols that are combined to create an image. For example ";-)" is the smiley emoticon and the following drawing represents a cow (skip cow [p. 24]):



Authoring tool

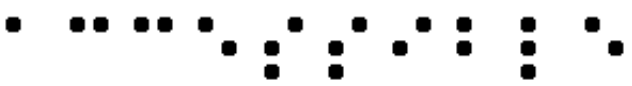
HTML editors, document conversion tools, tools that generate Web content from databases. Refer to the "Authoring Tool Accessibility Guidelines" ([WAI-AUTOOLS] [p. 28]) for information about developing accessible tools.

Backward compatible

Something that has been designed to work with earlier versions of a language, program, etc.

Braille

Braille uses six raised dots in different patterns to represent letters and numbers to be read by people who are blind with their fingertips. The word "Accessible" in braille follows:



A ***braille display***, commonly referred to as a "dynamic braille display," raises or lowers dot patterns on command from an electronic device, usually a computer. The result is a line of braille that can change from moment to moment. Dynamic braille displays range in size from one cell (six or eight dots) to an eighty cell line. Displays with twelve to twenty cells per line are the most common.

Collate

When referring to transcripts, collating means combining the text version of the descriptions and the text transcript (captions) of the primary audio track into a single document to read like a script of the movie. In other words, the two documents are not combined but flow as a single document.

Content/Structure/Presentation

The content of a document refers to both what the document says and the bytes (text and markup) that make it up. The structure of a document is how it is organized logically (e.g., by chapter, with an introduction and table of contents, etc.) The presentation of a document is how the document is rendered (e.g., as print, as a two-dimensional graphical presentation, as an text-only presentation, as synthesized speech, as braille, etc.

Consider a header, for example. The content of the header is what the header says (e.g., "Sailboats") and how it is marked up (e.g., with an H2 element in HTML). In terms of structure, the header may be part of a chapter of the document. Finally, the presentation of the header might be a bold block text in the margin, a centered line of text, a title spoken with a certain voice style (like an aural font), etc.

Content developer

Someone who authors Web pages or designs Web sites.

Dynamic HTML (DHTML)

DHTML is the marketing term applied to a mixture of standards including HTML, style sheets [p. 26] , the Document Object Model [DOM1] [p. 28] [DOM1] and scripting. However, there is no W3C specification that formally defines DHTML. Most guidelines may be applicable to applications using DHTML, however the following guidelines focus on issues related to scripting, and style sheets: guideline 1, guideline 2, guideline 8, and guideline 9.

Equivalent and ***Description***

An *equivalent* complements a primary functionality with a second, essentially equal functionality. For instance, an equivalent for an image functioning as a visual navigation arrow would convey information about navigation. Equivalents are important to users who, because of disability, technological constraints, or other circumstances, cannot or choose not to access the primary functionality in a more conventional way. Equivalents must be provided for images, image maps, tables (through captions), and applets.

A *description* provides information about the visual appearance or sound of content. Descriptions are generally longer than equivalents and may be external to a document (referenced by attributes such as "longdesc" or by inline markup, called *description links*, or d-links). Descriptions should be provided for audio and video clips, complex tables, images where detail is required, etc. Descriptions may provide information about functionality, but if the information is short, an equivalent should be used instead.

A ***text equivalent*** or *text description* is expressed in actual characters, making the information available for visual, braille, or speech output.

A *non-text equivalent* or *non-text description* is expressed in other media. Examples of non-text equivalents include graphical, video, pre-recorded audio equivalents. An example of a video equivalents would be videos of sign language renderings of the content. Non-text equivalents are particularly important for non-readers with deafness.

Image

A graphical presentation.

Image map

An image that has been divided into regions with associated actions. Clicking on an active region causes an action to occur.

When a user clicks on an active region of a client-side image map, the user agent calculates in which region the click occurred and follows the link associated with that region. Clicking on an active region of a server-side image map causes the coordinates of the click to be sent to a server, which then performs some action.

Content developers can make client-side image maps accessible by providing device-independent access to the same links associated with the image map's regions. Client-side image maps allow the user agent to provide immediate feedback as to whether or not the user's pointer is over an active region.

Important

Information is important if understanding it in detail is necessary for the overall understanding of a document.

Navigation Mechanism

A navigation mechanism is any means by which a user can navigate a page or site. Some typical mechanisms include:

navigation bars

A navigation bar is a collection of links to the most important parts of a document or site.

site maps

A site map provides a global view of the organization of a page or site.

tables of contents

A table of contents generally lists (and links to) the most important sections of a document.

Personal Digital Assistant (PDA)

A PDA is a small, portable computing device. Most PDAs are used to track personal data such as calendars, contacts, and electronic mail. A PDA is generally a handheld device with a small screen that allows input from various sources.

Screen magnifier

A software program that magnifies a portion of the screen, so that it can be more easily viewed. Screen magnifiers are used primarily by individuals with low vision.

Screen reader

A software program that reads the contents of the screen aloud to a user. Screen readers are used primarily by individuals who are blind, screen readers can usually only read text that is printed, not painted, to the screen.

Style sheets

A style sheet is a set of statements that specify presentation of a document. Style sheets may have three different origins: they may be written by content providers, created by users, or built into user agents. In CSS ([CSS2] [p. 28]), the interaction of content provider, user, and user agent style sheets is called the *cascade*.

Presentation markup is markup that achieves a stylistic (rather than structuring) effect such as the B or I elements in HTML. Note that the STRONG and EM elements are not considered presentation markup since they convey information that is independent of a particular font style.

User agent

Software to access Web content, including desktop graphical browsers, text browsers, voice browsers, mobile phones, multimedia players, plug-ins, and assistive technologies such as screen readers and screen magnifiers.

Acknowledgments

WAI Markup Guidelines Working Group Co-Chairs:

Chuck Letourneau, Starling Access Services

Gregg Vanderheiden, Trace Research and Development

W3C Team contacts:

Judy Brewer and Daniel Dardailler

We wish to thank the following people who have contributed their time and valuable comments to shaping these guidelines:

Harvey Bingham, Kevin Carey, Chetz Colwell, Neal Ewers, Geoff Freed, Al Gilman, Larry Goldberg, Jon Gunderson, Eric G. Hansen, Phill Jenkins, Leonard Kasday, George Kerscher, Marja-Riitta Koivunen, Josh Krieger, Scott Luebking, William Loughborough, Murray Maloney, Charles McCathieNevile, MegaZone (Livingston Enterprises), Masafumi Nakane, Mark Novak, Charles Oppermann, Mike Paciello, David Pawson, Michael Pieper, Greg Rosmaita, Liam Quinn, Dave Raggett, T.V. Raman, Robert Savellis, Jutta Treviranus, Steve Tyler, Jaap van Lelieveld, and Jason White

The original draft of this document is based on "The Unified Web Site Accessibility Guidelines" ([UWSAG] [p. 29]) compiled by the Trace R & D Center at the University of Wisconsin. That document includes a list of additional contributors.

References

[CSS1]
"CSS, level 1 Recommendation", B. Bos, H. Wium Lie, eds. The CSS1 Recommendation is available at:
http://www.w3.org/TR/1999/REC-CSS1-19990111.

[CSS2]
"CSS, level 2 Recommendation", B. Bos, H. Wium Lie, C. Lilley, and I. Jacobs, eds. The CSS2 Recommendation is available at:
http://www.w3.org/TR/1998/REC-CSS2-19980512.

[DOM1]
"Document Object Model (DOM) Level 1 Specification", V. Apparao, S. Byrne, M. Champion, S. Isaacs, I. Jacobs, A. Le Hors, G. Nicol, J. Robie, R. Sutor, C. Wilson, and L. Wood, eds. The DOM Level 1 Recommendation is available at:
http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001.

[HTML40]
"HTML 4.0 Recommendation", D. Raggett, A. Le Hors, and I. Jacobs, eds. The HTML 4.0 Recommendation is available at:
http://www.w3.org/TR/1998/REC-html40-19980424.

[HTML32]
"HTML 3.2 Recommendation", D. Raggett, ed. The HTML 3.2 Recommendation is available at:
http://www.w3.org/TR/REC-html32.

[MATHML]
"Mathematical Markup Language", P. Ion and R. Miner, eds. The MathML 1.0 Recommendation is available at:
http://www.w3.org/TR/1998/REC-MathML-19980407.

[PNG]
"PNG (Portable Network Graphics) Specification", T. Boutell, ed., T. Lane, contributing ed. The PNG Recommendation is available at:
http://www.w3.org/TR/REC-png.

[RDF]
"Resource Description Framework (RDF) Model and Syntax Specification", O. Lassila, R. Swick, eds. The RDF Recommendation is available at:
http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

[SMIL]
"Synchronized Multimedia Integration Language (SMIL) 1.0 Specification", P. Hoschka, editor. The SMIL 1.0 Recommendation is available at:
http://www.w3.org/TR/1998/REC-smil-19980615

[WAI-AUTOOLS]
"Authoring Tool Accessibility Guidelines", J. Treviranus, J. Richards, I. Jacobs, C. McCathieNevile, eds. The latest Working Draft of these guidelines for designing accessible authoring tools is available at:
http://www.w3.org/TR/WD-WAI-AUTOOLS/

[WAI-USERAGENT]
"User Agent Accessibility Guidelines", J. Gunderson and I. Jacobs, eds. The latest Working Draft of these guidelines for designing accessible user agents is available at:
http://www.w3.org/TR/WD-WAI-USERAGENT/

[UWSAG]
"The Unified Web Site Accessibility Guidelines", G. Vanderheiden, W. Chisholm, eds. The Unified Web Site Guidelines were compiled by the Trace R & D Center at the University of Wisconsin under funding from the National Institute on Disability and Rehabilitation Research (NIDRR), U.S. Dept. of Education. This document is available at:
http://www.tracecenter.org/docs/html_guidelines/version8.htm

[XML]
"Extensible Markup Language (XML) 1.0.", T. Bray, J. Paoli, C.M. Sperberg-McQueen, eds. The XML 1.0 Recommendation is available at:
http://www.w3.org/TR/1998/REC-xml-19980210