W3C

Web Content Accessibility Guidelines 1.0

W3C Proposed Recommendation 24-Mar-1999

This version:
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324
Latest version:
http://www.w3.org/TR/WAI-WEBCONTENT
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 1.0
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

These guidelines explain how to make Web content [p. 24] accessible to people with disabilities. The guidelines are intended for all Web content developers [p. 25] (page authors and site designers) and for developers of authoring tools [p. 24] . The primary goal of these guidelines is to promote accessibility. However, following them will also make Web content more available to *all* users, whatever user agent [p. 27] they are using (e.g., desktop browser, voice browser, mobile phone, automobile-based PC, etc.) or constraints they may be operating under (e.g., 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 content more accessible to a wide audience.

This document is part of a series of accessibility guidelines published by the Web Accessibility Initiative. The series also includes User Agent Accessibility Guidelines ([WAI-USERAGENT] [p. 30]) and Authoring Tool Accessibility Guidelines ([WAI-AUTOOLS] [p. 30]).

This is a reference document for accessibility principles and design ideas. It is meant to be stable and therefore does not provide specific information about browser support for different technologies as that information changes rapidly. Instead, the Web Accessibility Initiative Web site provides such information.

Status of this document

This document is a Proposed Recommendation of the World Wide Web Consortium (W3C) and is currently undergoing review by the Members of the W3C. Review comments on this specification should be sent by 21 April 1999 to w3c-wai-gl@w3.org. An archive of public comments is available. W3C Members may send their formal comments, visible only to the W3C Team, to w3c-wai-wcag-review@w3.org.

This specification is a revision of the last call public Working Draft dated 26 February 1999. It incorporates suggestions from reviewers and further deliberations of the Web Content Guidelines Working Group. A detailed list of changes to the document is available from the Web Content Guidelines Working Group site.

The Working Group anticipates no further substantial changes to this specification. We encourage active implementation to test this specification during the Proposed Recommendation review period.

Publication as a Proposed Recommendation does not imply endorsement by the W3C membership. This is still a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite W3C Drafts as other than "work in progress."

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

This document has been produced as part of the W3C Web Accessibility Initiative. The goal of the Web Content Guidelines Working Group is discussed in the Working Group charter.

Related documents

The current document includes an appendix entitled "List of Checkpoints for the Web Content Accessibility Guidelines 1.0." 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), 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:

- pointers on where to find information on browser support for the techniques.
- techniques for document validation and testing
- a techniques index of HTML elements and attributes that indicates which version of HTML defines them, which are deprecated [p. 25] in HTML 4.0, etc.
- a checkpoint map indicating where checkpoints are 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/TR/1999/WAI-WEBCONTENT-19990324/wai-pageauth.html
A plain text file:
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324/wai-pageauth.txt,
HTML (Guidelines, List of Checkpoints, Techniques) as a gzip'ed tar file :
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324/wai-pageauth.tgz,
HTML (Guidelines, List of Checkpoints, Techniques) as a zip archive:
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324/wai-pageauth.zip,
A PostScript file:
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324/wai-pageauth.ps,
A PDF file:
http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324/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
1. Document conventions 7
4. Priorities 7
5. Conformance 8
6. Web Content Accessibility Guidelines 9
1. Provide equivalent alternatives to auditory and visual content. . . .9
2. Don’t rely on color alone. 10
3. Use markup and style sheets properly. 11
4. Clarify natural language usage 12
5. Create tables that transform gracefully. 13
6. Ensure that pages featuring new technologies transform gracefully. 14
7. Ensure user control of time-sensitive content changes. . . . 15
8. Ensure direct accessibility of embedded user interfaces. . . . 16
9. Design for device-independence. 16
10. Use interim solutions. 17
11. Use W3C technologies and guidelines. 18
12. Provide context and orientation information. 19
13. Provide clear navigation mechanisms. 20
14. Ensure that documents are clear and simple. 21
Appendix A. - Validation 22
Appendix B. - Glossary 23
Acknowledgments 28
References 29

Additional appendix: List of Checkpoints for the Web Content Accessibility Guidelines 1.0.

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, move, or 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]). Text can be rendered in ways that are available to almost all browsing devices and accessible to almost all users.
- Create documents that 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 11.

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 mechanisms for navigating within and between pages. Providing navigation tools and orientation information in pages will maximize accessibility and usability. 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 12 to 14.

3. How the Guidelines are Organized

This document includes fourteen guidelines, or general principles of accessible design. Each guideline includes:

- The guideline number.
- The statement of the guideline.
- Guideline navigation links. Three links allow navigation to the next guideline (right arrow icon), the previous guideline (left arrow icon), or the current guideline's position in the table of contents (up arrow icon).
- The rationale behind the guideline and some groups of users who benefit from it.
- A list of checkpoint definitions.

The list of checkpoint definitions in each guideline explain how the guideline applies in typical content development scenarios. Each checkpoint definition includes:

- The checkpoint number.
- The statement of the checkpoint.
- The priority of the checkpoint. Priority 1 labels are highlighted through style sheets.
- Optional informative notes, clarifying examples, and cross references to related guidelines or checkpoints.
- A link 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.

1. Document conventions

The following editorial conventions are used throughout this document:

- Element names are in uppercase letters.
- Attribute names are quoted in lowercase letters.
- Links to definitions are rendered specially through style sheets.

4. Priorities

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

[Priority 1]
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]
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]
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 section defines three levels of conformance to this document:

- **Conformance Level "A"**: all Priority 1 checkpoints are satisfied;
- **Conformance Level "Double-A"**: all Priority 1 and 2 checkpoints are satisfied;
- **Conformance Level "Triple-A"**: all Priority 1, 2, and 3 checkpoints are satisfied;

Note. Conformance levels are spelled out in text so they may be understood when rendered to speech.

Claims of conformance to this document must use one of the following two forms.

Form 1: Specify:

- The guidelines title: "Web Content Accessibility Guidelines 1.0"
- The guidelines URI: http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324
- The conformance level satisfied: "A", "Double-A", or "Triple-A".
- The scope covered by the claim (e.g., page, or a defined scope of site).

Example of Form 1:

"This page conforms to W3C's "Web Content Accessibility Guidelines 1.0", available at http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990324, level Double-A."

Form 2: Include, on each page claiming conformance, one of three icons provided by W3C and link the icon to the appropriate W3C explanation of the claim.

Note. *In the event this document becomes a Recommendation, by that date WAI will provide a set of three icons, for "A", "Double-A", or "Triple-A" conformance levels of "Web Content Accessibility Guidelines 1.0", together with a stable URI to the W3C Web site for linking the icons to the W3C explanation of conformance claims.*

6. Web Content Accessibility Guidelines

Guideline 1. Provide equivalent alternatives to auditory and visual content.

Provide content that, when presented to the user, conveys essentially the same function or purpose as auditory or visual content.

Although some people cannot use images, movies, sounds, applets, etc. directly, they may still use pages that include equivalent [p. 25] alternatives to the visual or auditory content. The equivalent information must serve the same purpose as the visual or auditory content. Thus, a text equivalent for an image of an upward arrow that links to a table of contents could be "Go to table of contents". In some cases, an equivalent should describe the appearance of visual content or the sound of auditory content as well (e.g., for complex charts, billboards, or diagrams).

This guideline emphasizes the importance of providing text equivalents [p. 25] of non-text media (images, pre-recorded audio, video) because text equivalents can be rendered in so many ways (e.g., braille, synthesized speech, visually-displayed text). If text equivalents are not provided for visual and auditory information, people who are blind, have low vision, who are deaf or hard of hearing, or any user who cannot or has chosen not to use visual or auditory information will not know the purpose of those elements of the page.

Providing non-text equivalents (pictures, videos, audio) of text is also beneficial to some users, especially nonreaders or people who have difficulty reading. 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.

For an audio description of a visual presentation (e.g., movie) that already has an auditory track:

- synchronize the audio description with the auditory track,
- collate [p. 24] a text version of the audio description with a text version of the auditory track (caption). The collated text 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.

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

Checkpoints:

1.1 Provide a text equivalent for every non-text element (e.g., via "alt", "longdesc", or in element content). *This includes*: images, graphical representations of text, image map regions, short animations (e.g., animated GIFs), applets, ascii art, frames, scripts, inserted list bullets, sounds (played with or without user interaction), stand-alone audio files, synthesized speech, audio tracks of video, and video. [Priority 1]

For example, in HTML:

- Use "alt" for the IMG, INPUT, and APPLET elements, or provide equivalent information in the content of the OBJECT and APPLET elements.
- For complex elements where the "alt" text does not suffice, provide an additional description using, for example, "longdesc" with IMG or FRAME, a link inside an OBJECT element, or a description link [p. 25] in the document.
- For image maps, use the "alt" attribute with AREA, or the MAP element with A elements and rich description as content.

Techniques for checkpoint 1.1

1.2 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. Redundant text links for client-side image maps are only required until user agents [p. 27] render text equivalents for the map links.]

Techniques for checkpoint 1.2

1.3 For each movie, provide an auditory description of the video track and synchronize it with the audio track. [Priority 1]

Techniques for checkpoint 1.3

1.4 For any time-based presentation (e.g., a movie, animation, or multimedia presentation), synchronize equivalent alternatives (e.g., captions or video descriptions) with the presentation. [Priority 1]

Techniques for checkpoint 1.4

1.5 Replace ASCII art with an image or explain it. [Priority 1 or Priority 2 depending on the importance of the information.]

Refer also to checkpoint 1.1, checkpoint 13.10, and the example of ascii art in the glossary [p. 23] .

Techniques for checkpoint 1.5

Guideline 2. 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:

2.1 Ensure that all information conveyed with color is also available without color, for example from context or markup. [Priority 1]
Techniques for checkpoint 2.1
2.2 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 for images, Priority 3 for text].
Techniques for checkpoint 2.2

Guideline 3. 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 structure (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 appropriate markup because a certain browser or assistive technology does not process it correctly. For example, it is appropriate to use the TABLE element in HTML to mark up tabular information. [p. 27] Doing so (and creating tables that transform gracefully (refer to guideline 5) make it possible for software to render tables other than as two-dimensional grids. Refer also to checkpoint 10.3.

Checkpoints:

3.1 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. 19] Refer also to guideline 6 and guideline 11.
Techniques for checkpoint 3.1

3.2 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 3.2

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

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

Techniques for checkpoint 3.3

3.4 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 3.4

3.5 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 3.5

3.6 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 3.6

3.7 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. If absolute units are used, validate that the rendered content is usable (refer to the section on validation [p. 22]).

Techniques for checkpoint 3.7

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

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

Techniques for checkpoint 3.8

Guideline 4. 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 [p. 26] of a document's content 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, or people who are deaf or hard of hearing.

Checkpoints:

4.1 Clearly identify changes in the natural language of a document's text and any text equivalents [p. 25] (e.g., captions) of non-text content. [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 4.1
4.2 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 4.2
4.3 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 4.3

Guideline 5. 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 information [p. 27] ("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] ([-linear-tables]).

Many user agents [p. 27] transform tables to present them and if not marked up properly, the tables will not make sense when rendered. Refer also to guideline 3.

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:

5.1 For data tables, identify row and column headers. [Priority 1]
For example, in HTML, use TD to identify data cells and TH to identify headers.
Techniques for checkpoint 5.1
5.2 For data tables that have two or more logical levels of row or column headers, 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 5.2

5.3 Avoid using tables for layout. [Priority 2]
Techniques for checkpoint 5.3

5.4 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.
Techniques for checkpoint 5.4

5.5 Provide summaries for tables. [Priority 3]
For example, in HTML, use the "summary" attribute of the TABLE element.
Techniques for checkpoint 5.5

5.6 Provide abbreviations for header labels. [Priority 3]
For example, in HTML, use the "abbr" attribute on the TH element.
Techniques for checkpoint 5.6

Refer also to checkpoint 10.3 and checkpoint 3.1.

Guideline 6. 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:

6.1 Organize content logically using appropriate structural markup so the organization remains clear even when associated style sheets are turned off or are not supported. [Priority 1]
This makes it more likely that the document will be understood even when styles are turned off or overridden by the user.
Techniques for checkpoint 6.1

6.2 Ensure that descriptions and text alternatives for dynamic content are updated when the dynamic content changes. [Priority 1]
Techniques for checkpoint 6.2

6.3 Ensure that pages are usable when scripts, applets, or other programmatic objects are turned off or not supported. If this is not possible, provide equivalent mechanisms on an alternative accessible page. [p. 19] [Priority 1]
For example, in HTML provide a text equivalent [p. 25] with the NOSCRIPT element or via a server-side script. Or provide a non-text equivalent (e.g., a snapshot in place of an animation, a video equivalent of an applet, etc.). For applets and programmatic objects, provide text equivalents [p. 25] . Refer also to guideline 1.

Techniques for checkpoint 6.3

6.4 For scripts and applets, until user agents [p. 27] provide device-independent means to activate event handlers, ensure that event handlers are keyboard operable. [Priority 2]

Techniques for checkpoint 6.4

6.5 Provide an alternative presentation or page when the primary content is dynamic (e.g., when frame contents change, when scripts cause changes, etc.). [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 6.5

Refer also to checkpoint 11.4.

Guideline 7. 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.

Note. All of the following checkpoints involve some content developer responsibility until user agents [p. 27] provide adequate feature control mechanisms.

Checkpoints:

7.1 Until user agents [p. 27] allow users to control it, avoid causing the screen to flicker. [Priority 1]

Note. 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).

Techniques for checkpoint 7.1

7.2 Until user agents [p. 27] allow users to control it, avoid causing content to blink (i.e., change presentation at a regular rate, such as turning on and off). [Priority 2]

Techniques for checkpoint 7.2

7.3 Until user agents [p. 27] allow users to freeze moving content, avoid movement in pages. [Priority 2]

When a page includes moving content, provide a mechanism within a script or applet to allow users to freeze motion or updates. Using style sheets with scripting to create movement allows users to turn off or override the effect more easily. Refer also to guideline 8.

Techniques for checkpoint 7.3

7.4 Until user agents [p. 27] provide the ability to stop the refresh, do not create periodically auto-refreshing pages. [Priority 2]
For example, in HTML, don't cause pages to auto-refresh with "HTTP-EQUIV=refresh" until user agents allow users to turn off the feature.
Techniques for checkpoint 7.4
7.5 Until user agents [p. 27] 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 7.5

Note. The BLINK and MARQUEE elements are not defined in any W3C HTML specification and should not be used. Refer also to guideline 11.

Guideline 8. 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. 30]), the Authoring Tool Accessibility Guidelines ([WAI-AUTOOL] [p. 30]), and the discussion of applets and other objects in the Techniques Document.

Checkpoint:

8.1 Make programmatic elements such as scripts and applets directly accessible or compatible with assistive technologies [Priority 1 if functionality is important [p. 26] and not presented elsewhere, otherwise Priority 2.]
Refer also to guideline 6.
Techniques for checkpoint 8.1

Guideline 9. 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 such as 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 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:

9.1 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]
Refer also to checkpoint 1.1 and checkpoint 1.2.
Techniques for checkpoint 9.1
9.2 Ensure that all elements that have their own interface are keyboard operable. [Priority 2]
Refer also to guideline 8.
Techniques for checkpoint 9.2
9.3 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 9.3
9.4 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 9.4
9.5 Provide keyboard shortcuts to important 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 9.5

Guideline 10. 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 user agents [p. 27] and assistive technologies [p. 23] address these issues.

Checkpoints:

10.1 Until user agents [p. 27] allow users to turn off spawned windows, 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 10.1

10.2 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). Refer also to checkpoint 12.4.

Techniques for checkpoint 10.2

10.3 Until user agents [p. 27] or assistive technologies render side-by-side text correctly, 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. 27] (such as some screen readers [p. 26]) that are unable to handle blocks of text presented side-by-side; the checkpoint should not discourage content developers from using tables to represent tabular information [p. 27] .

Techniques for checkpoint 10.3

10.4 Until user agents [p. 27] handle empty controls correctly, 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 10.4

10.5 Until user agents [p. 27] or assistive technologies render adjacent links distinctly, include non-link, printable characters (surrounded by spaces) between adjacent links. [Priority 3]

Techniques for checkpoint 10.5

Guideline 11. 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 formats (e.g., PDF, Shockwave, etc.) require viewing with either plug-ins or stand-alone applications. Often, these formats cannot be viewed or navigated with 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, validate each page for accessibility and usability after the conversion process (refer to the section on validation [p. 22]). 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:

11.1 Use W3C technologies and use the latest versions when they are supported. [Priority 2]
Refer to the list of references [p. 29] for information about where to find the latest W3C specifications.
Techniques for checkpoint 11.1
11.2 Avoid deprecated features of W3C technologies. [Priority 2]
For example, in HTML, don't use the deprecated [p. 25] FONT element; use style sheets instead (e.g., the 'font' property in CSS).
Techniques for checkpoint 11.2
11.3 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 11.3
11.4 If, after best efforts [p. 19] , you cannot create an accessible page, provide a link to an alternative page that uses W3C technologies, is accessible, has equivalent [p. 25] information, and is updated as often as the inaccessible (original) page. [Priority 1]
Techniques for checkpoint 11.4

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 unavailable. 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 12. 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:

12.1 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 12.1
12.2 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 12.2
12.3 Divide large blocks of information into more manageable groups where natural and appropriate. [Priority 2]
For example, in HTML, use OPTGROUP to group OPTION elements inside a SELECT; group form controls with FIELDSET and LEGEND; use nested lists where appropriate; use headings to structure documents, etc. Refer also to guideline 3.
Techniques for checkpoint 12.3
12.4 Associate labels explicitly with their controls. [Priority 2]
For example, in HTML use LABEL and its "for" attribute.
Techniques for checkpoint 12.4

Guideline 13. 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] are important to people with cognitive disabilities or blindness, and benefit all users.

Checkpoints:

13.1 Clearly identify the target of each link. [Priority 2]
Link text [p. 26] should be meaningful enough to make sense when read out of context - either on its own or as part of a sequence of links. Link text should also be terse.
For example, in HTML, write "Information about version 4.3" instead of "click here". In addition to clear link text, content developers may further clarify the target of a link with an informative link title (e.g., in HTML, the "title" attribute).
Techniques for checkpoint 13.1

13.2 Provide metadata to add semantic information to pages and sites. [Priority 2]
For example, use RDF ([RDF] [p. 29]) to indicate the document's author, the type of content, and to describe the navigation mechanism, etc.
Note. Some HTML user agents [p. 27] 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.). Refer also to checkpoint 13.5.
Techniques for checkpoint 13.2
13.3 Provide information about the general layout of a site (e.g., a site map, or table of contents). [Priority 2]
Also indicate what accessibility features are available and how to use them.
Techniques for checkpoint 13.3
13.4 Use navigation mechanisms in a consistent manner. [Priority 2]
Techniques for checkpoint 13.4
13.5 Provide navigation bars to highlight and give access to the navigation mechanism. [Priority 3]
Techniques for checkpoint 13.5
13.6 Group related links, identify the group (for user agents), and, until user agents [p. 27] do so, provide a way to bypass the group. [Priority 3]
Techniques for checkpoint 13.6
13.7 Enable different types of searches for different skill levels and preferences. [Priority 3]
Techniques for checkpoint 13.7
13.8 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 13.8
13.9 Provide information about document collections (i.e., documents comprising multiple pages.). [Priority 3]
For example, in HTML describe document collections with the LINK element and the "rel" and "rev" attributes. Another way to create a collection is by building an archive (e.g., with zip, gzip, stuffit, etc.) of the multiple pages.
Note. The performance improvement gained by offline processing can make browsing much less expensive for people with disabilities who may be browsing slowly.
Techniques for checkpoint 13.9
13.10 Provide a means to skip over multi-line ASCII art. [Priority 3]
Refer also to checkpoint 1.5.
Techniques for checkpoint 13.10

Guideline 14. 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 users. 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. Access to written information can be difficult to impossible for people who have cognitive disabilities, learning disabilities, or who are deaf or hard of hearing. This consideration also applies to the many people whose first language differs from your own.

Checkpoints:

14.1 Use the clearest and simplest language appropriate for a site's content. [Priority 1]
Techniques for checkpoint 14.1
14.2 Provide visual or auditory equivalents to text where they facilitate comprehension of the page. [Priority 3]
Note. For example, provide a video representation of an applet, an audio description of video, recorded spoken text, etc. These equivalent presentations are particularly important for non-readers. This checkpoint does not mean that text content should be removed. Refer also to guideline 1.
Techniques for checkpoint 14.2
14.3 Create a style of presentation that is consistent across pages. [Priority 3]
Techniques for checkpoint 14.3

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 text equivalent [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. - Glossary

Accessible
Content is accessible when it may be used by someone with a disability.

Applet
A program inserted into a Web page.

Assistive technology
Software that provides a specialized service and generally operates in conjunction with an independent user agent [p. 27] . Assistive technologies include screen readers, screen magnifiers, and voice input software.

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 ascii cow [p. 24]]:

```
          (__)
          (oo)
    /-------\/
   / |     ||
  *  ||----||
```

Authoring tool

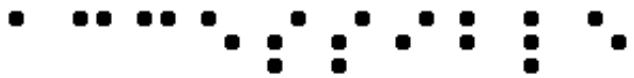
HTML editors, document conversion tools, tools that generate Web content from databases. Refer to the "Authoring Tool Accessibility Guidelines" ([WAI-AUTOOLS] [p. 30]) for information about developing accessible tools.

Backward compatible

Design that continues 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.

Caption

When a text equivalent [p. 25] for video is synchronized with the video presentation it is called a *caption*. Captions are used by people who cannot hear the audio track of the video material. Without 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.

Collate

When referring to text transcripts [p. 25] , 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.

Deprecated

A deprecated element or attribute is one that has been outdated by newer constructs. Deprecated elements may become obsolete in future versions of HTML. The techniques index of HTML elements and attributes in the Techniques Document indicates which elements and attributes are deprecated in HTML 4.0.

Authors should avoid using deprecated elements and attributes. User agents should continue to support for reasons of backward compatibility.

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. 29] 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 6, and guideline 7.

Equivalent

An "equivalent" is information that, when presented to a user, provides essentially the same function (or fulfills the same purpose) as another piece of information. Due to the importance of text for a variety of output devices, we distinguish ***text equivalents*** from ***non-text equivalents***. For example, for an audio clip of a conversation between two people, a ***text-transcript*** is a text equivalent of the audio clip. Examples of non-text equivalents include spoken audio recordings of text, icons, and videos of sign language renderings of content. In the context of this document, text and non-text equivalents are generally intended to serve as alternatives for portions of the primary content that are inaccessible to people with disabilities.

Part of providing equivalent information may involve describing the appearance (visual or audio) of an image, table, chart, or other complex object. Descriptive information may be part of the equivalent or may be separated from it (often the case when the description is lengthy). Descriptive information may even be external to the document, available through a link (e.g., via the "longdesc" attribute in HTML or via a *description links*, or d-links).

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.

Link text

The rendered text content of a link.

Natural Language

National spoken or written languages such as French, English, and Japanese. The natural language of content may be indicated with the "lang" attribute in HTML ([HTML40] [p. 29] , section 8.1) and the "xml:lang" attribute in XML ([XML] [p. 30] , section 2.12).

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. 29]), 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.

Tabular information

When tables are used to represent logical relationships among data - text, numbers, images, etc., that information is called "tabular information" and the tables are called "data tables". The relationships expressed by a table may be rendered visually (usually on a two-dimensional grid), aurally (often preceding cells with header information), or in other formats.

Until user agents ...

In most of the checkpoints , content developers are asked to ensure the accessibility of their pages and sites. However, there are accessibility needs that would be more appropriately met by a user agent [p. 27] or assistive technology [p. 23] . Unfortunately, not all user agents or assistive technologies provide the accessibility control users require (e.g., some user agents may not allow users to turn off blinking content, or some screen readers may not handle tables well). Therefore, to address cases where the responsibility for providing accessible control lies with the user agent or assistive technology but that control is not yet easily or widely available, certain checkpoints contain the phrase "until user agents ...". When content developers see this phrase, they should recognize that they are being asked to provide additional support for accessibility until most user agents are capable of providing the necessary control.

For each checkpoint containing this phrase, content developers must consider:

1. What is their anticipated audience? For example, designing for a company intranet where everyone uses the same browser is different than designing for the World Wide Web.
2. Do those users have tools available that satisfy the demands of the checkpoint? If the answer to this question is yes, content developers are not required to satisfy the checkpoint.

How will content developers know when most user agents or assistive technologies meet specific needs? The W3C WAI will make information about support for accessibility features available from its Web site (either directly or by providing links to the information). Content developers are encouraged to consult this information regularly for pertinent updates.

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

Web Content 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 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. 30]) compiled by the Trace R & D Center at the University of Wisconsin. That document includes a list of additional contributors.

References

For the latest version of any W3C specification please consult the list of W3C Technical Reports.

[CSS1]
"CSS, level 1 Recommendation", B. Bos, H. Wium Lie, eds., 17 December 1996, revised 11 January 1999. The CSS1 Recommendation is available at: http://www.w3.org/TR/1999/REC-CSS1-19990111.
The latest version of CSS1 is available at: http://www.w3.org/TR/REC-CSS1.

[CSS2]
"CSS, level 2 Recommendation", B. Bos, H. Wium Lie, C. Lilley, and I. Jacobs, eds., 12 May 1998. The CSS2 Recommendation is available at: http://www.w3.org/TR/1998/REC-CSS2-19980512.
The latest version of CSS2 is available at: http://www.w3.org/TR/REC-CSS2.

[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., 1 October 1998. The DOM Level 1 Recommendation is available at: http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001.
The latest version of DOM Level 1 is available at: http://www.w3.org/TR/REC-DOM-Level-1

[HTML40]
"HTML 4.0 Recommendation", D. Raggett, A. Le Hors, and I. Jacobs, eds., 17 December 1997, revised 24 April 1998. The HTML 4.0 Recommendation is available at: http://www.w3.org/TR/1998/REC-html40-19980424.
The latest version of HTML 4.0 is available at: http://www.w3.org/TR/REC-html40.

[HTML32]
"HTML 3.2 Recommendation", D. Raggett, ed., 14 January 1997. The latest version of HTML 3.2 is available at: http://www.w3.org/TR/REC-html32.

[MATHML]
"Mathematical Markup Language", P. Ion and R. Miner, eds., 7 April 1998. The MathML 1.0 Recommendation is available at: http://www.w3.org/TR/1998/REC-MathML-19980407.
The latest version of MathML 1.0 is available at: http://www.w3.org/TRREC-MathML.

[PNG]
"PNG (Portable Network Graphics) Specification", T. Boutell, ed., T. Lane, contributing ed., 1 October 1996. The latest version of PNG 1.0 is available at: http://www.w3.org/TR/REC-png.

[RDF]
"Resource Description Framework (RDF) Model and Syntax Specification", O. Lassila, R. Swick, eds., 22 February 1999. The RDF Recommendation is available at: http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

The latest version of RDF 1.0 is available at:
http://www.w3.org/TR/REC-rdf-syntax

[SMIL]

"Synchronized Multimedia Integration Language (SMIL) 1.0 Specification", P. Hoschka, ed., 15 June 1998. The SMIL 1.0 Recommendation is available at: http://www.w3.org/TR/1998/REC-smil-19980615
The latest version of SMIL 1.0 is available at: http://www.w3.org/TR/REC-smil

[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., 10 February 1998. The XML 1.0 Recommendation is available at: http://www.w3.org/TR/1998/REC-xml-19980210.
The latest version of XML 1.0 is available at: http://www.w3.org/TR/REC-xml