User Agent Accessibility Guidelines 1.0

W3C Working Draft 16-July-1999

This version:
http://www.w3.org/WAI/UA/WAI-USERAGENT-19990716
(plain text, postscript, pdf, gzip tar file of HTML, zip archive of HTML)
Latest version:
http://www.w3.org/WAI/UA/WAI-USERAGENT
Previous version:
http://www.w3.org/WAI/UA/WAI-USERAGENT-19990709

Editors:
Jon Gunderson <jongund@uiuc.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 user agent manufacturers for making their products more accessible to people with disabilities and for increasing usability for all users. This document emphasizes the accessibility of interoperability between two important classes of user agents - graphical desktop browsers and dependent assistive technologies (screen readers, screen magnifiers, braille displays, and voice input software). However, it is meant to be applicable to user agents in general, including text and voice browsers, multimedia players, and plug-ins.

This document includes an appendix that organizes all of the checkpoints [p. 7] by topic and priority. The checkpoints in the appendix link to their definitions in the current document. The topics identified in the appendix include user interface, keyboard support, images, multimedia, tables, frames, forms, and scripts. This appendix is available both as a tabular summary and as a simple list of checkpoints.

A separate document, entitled "Techniques for User Agent Accessibility Guidelines 1.0" ([UA-TECHNIQUES] [p. 25]), explains how to implement the checkpoints defined in the current document. The Techniques Document discusses each checkpoint in more detail and provides examples using the Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), Synchronized Multimedia Integration Language (SMIL), and the Mathematical Markup Language (MathML). The Techniques Document has been designed to track changes in technology and is

expected to be updated more frequently than the current document. **Note.** Some browsers or multimedia tools may not support some of the features described in the guidelines. In particular, new features of HTML 4.0 or CSS 1 or CSS 2 may not be supported.

"User Agent Accessibility Guidelines 1.0" is part of a series of accessibility guidelines published by the Web Accessibility Initiative. The series also includes Web Content Accessibility Guidelines ([WAI-WEBCONTENT] [p. 26]) and Authoring Tool Accessibility Guidelines ([WAI-AUTOOLS] [p. 25]).

Status of this document

This is a W3C Working Draft for review by W3C Members and other interested parties. It 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 User Agent (UA) Working Group.

Please send comments about this document to the public mailing list: w3c-wai-ua@w3.org.

This document has been produced as part of the Web Accessibility Initiative, and is intended as a draft of a Proposed Recommendation for how to improve user agent accessibility. The goals of the WAI UA Working Group are discussed in the WAI UA charter. A list of the current Working Group members is available.

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

Table of Contents

1. Introduction 4
    1.1 Principles of Accessible Design 5
2. How the Guidelines are Organized 6
    2.1 Document conventions 7
3. Priorities 7
    3.1 Conformance 8
4. User Agent Accessibility Guidelines 1.0
    1. Support input and output device-independence 1.0
    2. Ensure keyboard access to user agent functionalities 1.0
    3. Provide accessible product documentation and help 1.1
    4. Allow the user to configure the user agent 1.2
    5. Allow the user to turn off features that may reduce accessibility . 1.2
    6. Ensure user control over document styles 1.3
    7. Ensure user access to document content 1.4
    8. Provide navigation mechanisms 1.6
    9. Help orient the user 1.6
    10. Notify the user of document changes 1.8
    11. Support applicable W3C technologies and guidelines . . . 1.9
    12. Observe system conventions and standard interfaces . . . 1.9
Appendix A. -- Glossary 20
5. Acknowledgments 24
6. References 25

The appendix list of checkpoints is available as either a tabular summary of checkpoints or as a simple list of checkpoints.

1. Introduction

For those unfamiliar with accessibility issues pertaining to user agent design, consider that many users may be using documents 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.).

The guidelines in this document are designed to help developers understand and thereby reduce accessibility barriers that impede access to the Web. The associated Techniques Document ([UA-TECHNIQUES] [p. 25]) provides practical solutions based on existing and upcoming technologies. Though developers may believe that implementing accessibility features in their products is difficult or of limited use, considering accessibility during the design phase of a product leads to more flexible, manageable, and extensible software.

These guidelines include information relevant to a wide class of user agents: graphical desktop browsers, screen readers, speech synthesizers, multimedia players, text browsers, voice browsers, plug-ins, etc., with a particular focus on two classes of user agents:

1. Graphical desktop browsers [p. 8]
2. Dependent user agents, [p. 8] which rely on of other user agents for input and/or output. Dependent user agents include:
   - screen magnifiers, which are used by people with visual impairment to enlarge and change colors on the screen to improve readability of text and images.
   - screen readers, which are used by people who are blind or with reading disabilities to read textual information through speech or braille displays.
   - alternative keyboards, which are used by people with movement impairments to simulate the keyboard.
   - alternative pointing devices, which are used by people with movement impairments to simulate mouse pointing and button activations.

The guidelines emphasize interoperability between these two classes of user agents.

1.1 Principles of Accessible Design

This document is organized according to several principles that will improve the design of any type of user agent:

Ensure that the user interface is accessible.

The user must have access to the functionality offered by the user agent through its user interface. This includes the functionalities built into the tool as well as those offered via the document: links, form controls, applets, and other interactive elements.

The general topic of user interface accessibility for computer software exceeds the scope of this document. The Techniques Document contains some pointers to software accessibility guidelines for The guidelines do discuss important user interface topics such as device-independence, configurability, and accessible product documentation. Software developers should also remember that user interfaces must be intuitive, simple, and tested. Features that are known to promote accessibility should be made obvious to users and easy to find.

Ensure that the user has access to content.

Through the user interface, users must have access to document content, including alternative content [p. 20] (e.g., "alt" attribute text in HTML). To ensure access to content, users must have final control of the style (colors, fonts, speech rate, speech volume, etc.) and format of a document.

User agents may facilitate access to content by providing a convenient user interface (e.g., scrolled viewport) and navigation mechanisms (e.g., allowing users to tab to active elements [p. 21] of a document).

Help orient the user.

Orientation - the sense of where one is within a document or series of documents - is fundamental to a successful Web experience. Visual mechanisms that promote orientation include:

- proportional scroll bars, which indicate how much of the document has been read
- frames, which suggest content relationships
- graphical site maps, which describe page organization

Some users cannot use visual orientation clues and require other mechanisms to remain oriented:

- Contextual information about document elements. For instance, users with blindness who navigate by surfing only the links of a document benefit from knowing how many links the document contains or the number of the current link. Numerical position information, in conjunction with a navigation mechanism that allows users to jump directly to a given link, can greatly facilitate browsing,

especially for documents with many links.
- Contextual information about the document or views (e.g., how many views, whether the document has finished loading, whether the user can begin to browse, etc.)
- Summary information about specific elements (e.g., the dimensions of a table).
- Alerts when events occur during browsing (e.g., a window opens, a script is executed, etc.).

The user agent should also minimize the chances the user will become disoriented by allowing the user to control when windows are spawned, warning the user when events occur, avoiding sudden movements of the viewport, etc.

Follow system standards and conventions.

Following system standards and conventions promotes accessibility in a number of ways:

- Observing system conventions in user interface design, software installation, and software documentation improves usability.
- Using standard system interfaces makes it possible for assistive technologies to access information predictably.

Communication through standard interfaces is particularly important for graphical desktop user agents, which must make information available to assistive technologies. Even when a user agent implements a feature natively, it should make information available to other software. Scripting tools and automated test engines also benefit from having access to user agent information through standard interfaces.

Implementing interoperable specifications (such as those produced by W3C) promotes accessibility for several reasons:

- W3C technologies include "built-in" accessibility features.
- W3C specifications undergo early review to ensure that accessibility issues are considered during the design phase.
- W3C specifications are developed in an open, industry consensus process.

2. 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 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 checkpoints are highlighted through the use of style sheets.
- Optional informative notes, clarifying examples, and cross references to related guidelines or checkpoints.
- A link to a section of the Techniques Document ([UA-TECHNIQUES] [p. 25]) where implementations and examples of the checkpoint are discussed.

Each checkpoint is intended to be specific enough so that someone reviewing a user agent may verify that the checkpoint has been satisfied.

2.1 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 highlighted through the use of style sheets.

3. Priorities

Each checkpoint in this document is assigned a priority that indicates its importance for users.

[Priority 1]
This checkpoint **must** be implemented by user agents as a native feature or through compatibility with assistive technology, otherwise one or more groups of users with disabilities will find it impossible to access information. Satisfying this checkpoint is a basic requirement for some individuals to be able to use the Web.

[Priority 2]
This checkpoint **should** be implemented by user agents as a native feature or through compatibility with assistive technology, otherwise one or more groups of users will find it difficult to access information. Satisfying this checkpoint will remove significant barriers to accessing Web documents.

[Priority 3]
This checkpoint **may** be implemented by user agents as a native feature or through compatibility with assistive technology, to make it easier for one or more groups of users to access information. Satisfying this checkpoint will improve access to the Web for some individuals.

3.1 Conformance

The terms "must", "should", and "may" (and related terms) are used in this document in accordance with RFC 2119 ([RFC2119] [p. 25]).

To promote interoperability between desktop graphical user agents [p. 8] and dependent user agents [p. 8] , conformance to this document is expressed in terms of these two types of software.

Conformance for desktop graphical user agents

In order to conform as a desktop graphical user agent, the user agent must do all of the following:

1. Satisfy all the checkpoints (for a chosen conformance level [p. 8]) that apply [p. 20] to desktop graphical user agents, and
2. Satisfy each checkpoint natively [p. 21] unless the checkpoint explicitly indicates that it may be satisfied through communication with other software.

Even for those checkpoints that must be satisfied natively, desktop graphical user agents should make information available to other software through standard interfaces (e.g., specialized dependent user agents may provide a better solution to a problem than a desktop user agent).

Conformance for dependent user agents

In order to conform as a dependent user agent, the user agent must do all of the following:

1. Satisfy all the checkpoints (for a chosen conformance level [p. 8]) that apply [p. 20] to dependent user agents, and
2. Satisfy each checkpoint natively [p. 21] unless the checkpoint explicitly indicates that it may be satisfied through communication with other software.

Conformance claims

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: "User Agent Accessibility Guidelines 1.0"
- The guidelines URI: http://www.w3.org/WAI/UA/WAI-USERAGENT-19990716
- The conformance level satisfied: "A", "Double-A", or "Triple-A".
- The class of user agent covered by the claim: desktop graphical user agent or dependent user agent.

Example of Form 1:

This product conforms as a desktop graphical user agent to W3C's "User Agent Accessibility Guidelines 1.0", available at http://www.w3.org/WAI/UA/WAI-USERAGENT-19990716, level Double-A.

or:

This screen reader conforms as a dependent user agent to W3C's "User Agent Accessibility Guidelines 1.0", available at http://www.w3.org/WAI/UA/WAI-USERAGENT-19990716, level Double-A.

Form 2: Include, on product packaging or documentation, one of three icons provided by W3C and for Web documentation, link the icon to the appropriate W3C explanation of the claim.

Note. *In the event this document becomes a W3C Recommendation, information about the icons and how to use them will be available at the W3C Web site.*

4. User Agent Accessibility Guidelines

Guideline 1. Support input and output device-independence

Ensure that the user can interact with the user agent (and the document it renders) using the supported input and output devices of their choice.

Input devices include pointing devices, keyboards, braille devices, head wands, microphones, touch screens, and others. Output devices may include monitors, speech synthesizers, braille devices, and printers.

Please note that "device-independent support" does not mean that user agents must support every input or output device. User agents should offer redundant input and output mechanisms for those devices that are supported. For example, if a user agent supports keyboard and mouse input, users should be able to interact with all features using either the keyboard or the mouse.

1.1 Ensure that all functionalities offered by the user agent interface are available through all supported input devices. [Priority 1]
1.2 Ensure that the user can interact with all active elements [p. 21] of a document in a device-independent manner. [Priority 1]
1.3 Ensure that the user can install the user agent software in a device-independent manner. [Priority 1]
1.4 Ensure that the user can configure the user agent in a device-independent manner. [Priority 1]
1.5 Ensure that the user can access user agent documentation in a device-independent manner. [Priority 1]
1.6 Ensure that all messages to the user (e.g., warnings, errors, etc.) are available in an output device-independent manner using any supported output devices. [Priority 1]

Guideline 2. Ensure keyboard access to user agent functionalities

Ensure that the user has access to user agent functionalities through the keyboard since keyboard access is widely supported.

Although keyboard access may seem to contradict the previous guideline on device-independence, ensuring keyboard access to user agent functionality is important to accessibility since keyboard access is available to many users and is widely supported. Even when a user doesn’t use a physical keyboard, it is still possible to simulate keyboard events through software. This guideline is important for ensuring compatibility between desktop graphical user agents and dependent assistive technologies.

Checkpoints in this section do not apply to user agents (e.g., kiosks) that do not natively support keyboard input.

One area where configurability is important involves keyboard access to user agent functionality. Some users require single-key access, others require that keys activated in combination be physically close together, while others require that they be spaced physically far apart.

The more apparent the keyboard commands are to all users, the more likely it is that new users with disabilities will find them and use them. Refer also to checkpoint 9.25.

2.1 By default and without additional customization, ensure that all functionalities offered by the user agent are accessible using the keyboard. [Priority 1]

> **Note.** This checkpoint intends to ensure compatibility with assistive technologies that rely on keyboard input. Functionalities include being able to show, hide, resize and move windows or panes created by the user agent.

2.2 Provide documentation on default keyboard commands and include with user agent documentation and/or user help system. [Priority 1]

> Refer also to guideline 3.

2.3 Provide information to the user about the current keyboard configuration. [Priority 1]

> **Note.** For example, users should be able to find information about complex key combinations. Refer also to guideline 3.

2.4 Allow the user to configure the keystrokes used to activate user agent functionalities. Wherever possible, allow single key activation of functions. [Priority 2]

2.5 Allow the user to turn on and off author-specified keyboard configurations. [Priority 2]

2.6 Indicate the keyboard access method to activate a user agent function using platform conventions. [Priority 2]

2.7 Avoid default keyboard configurations that interfere with system conventions. [Priority 2]

> For example, the default configuration should not include "Control-Alt-Delete" on systems where that combination has special meaning to the operating system. In particular, default configurations should not interfere with the mobility access keyboard modifiers reserved for the operating system. Refer also to guideline 12.

2.8 Provide a default keyboard configuration for frequently performed operations. [Priority 3]

Guideline 3. Provide accessible product documentation and help

Ensure that the user can learn about software features, notably those that relate to accessibility.

Users who may not be able to access print material, including individuals with visual impairments, learning disabilities, or movement impairments, may be able to use accessible electronic documentation or documents in alternative hardcopy formats.

Documentation includes **all** product documentation, notably installation instructions, the help system, and all product manuals. Refer also to guideline 2 and checkpoint 12.6.

3.1 Ensure that all product documentation conforms to the Web Content Accessibility Guidelines. [Priority 1]
Refer to [WAI-WEBCONTENT [p. 26]].
3.2 Ensure that all user agent functionalities that promote accessibility are documented. [Priority 1]
For example, review the documentation or help system to ensure that it discusses the functionalities addressed by the checkpoints of this document.
3.3 Describe product features known to promote accessibility in a section of the product documentation. [Priority 2]

Guideline 4. Allow the user to configure the user agent

Subhead here.

Refer also to guideline 2. Refer also to checkpoint 9.25.

4.1 Allow the user to configure the user agent in named profiles that may be shared (by other users or software). [Priority 2]
Users must be able to select from among available profiles or no profile (i.e., the user agent default settings).
4.2 Allow the user to configure the visual arrangement of user interface controls. [Priority 3]

Guideline 5. Allow the user to turn off features that may reduce accessibility

Subhead here.

User agents are only expected to provide this control for content that it recognizes. [p. 22] Refer also to guideline 6.

5.1 Allow the user to turn on and off rendering of images. [Priority 1]
5.2 Allow the user to turn on and off rendering of background images. [Priority 1]
5.3 Allow the user to turn on and off rendering of video. [Priority 1]
5.4 Allow the user to turn on and off rendering of sound. [Priority 1]

5.5 Allow the user to turn on and off rendering of captions. [Priority 1]
5.6 Allow the user to turn on and off animated or blinking text. [Priority 1]
Note. This is particularly important for users with screen readers.
5.7 Allow the user to turn on and off animations and blinking images. [Priority 1]
5.8 Allow the user to turn on and off support for scripts and applets. [Priority 1]
Note. This is particularly important for scripts that cause the screen to flicker, since people with photosensitive epilepsy can have seizures triggered by flickering or flashing in the 4 to 59 flashes per second (Hertz) range.
5.9 Allow the user to turn on and off support for user style sheets. [Priority 1]
5.10 Allow the user to turn on and off support for author style sheets. [Priority 1]
5.11 Allow the user to turn on and off support for spawned windows. [Priority 1]
5.12 Allow the user to turn on and off rendering of frames. [Priority 2]
5.13 Allow the user to turn on and off author-specified page forwards that occur after a time delay and without user intervention. [Priority 3]
5.14 Allow the user to turn on and off automatic page refresh. [Priority 3]

Guideline 6. Ensure user control over document styles

Ensure that the user has control over the colors, fonts and other stylistic aspects of a document and can author styles and user agent default styles.

In order to access a document, some users may require that it be rendered in a manner other than what the author intended. Users who are color-deficient may not be able to perceive certain color combinations. Users with low vision may require large fonts. Users who are blind may require audio or tactile rendering. Users who are deaf may require captions for audio files.

User agents must therefore allow the user to control:

- The document's style (e.g., fonts, colors, aural parameters, etc.)
- The document's formatting: whether the document is presented textually, graphically, linearly, aurally, for tactile use, or some combination of these.
- The document's content. This means whether primary content or alternative representations of content or both are rendered.
- The user interface. Since authors may make changes to the user interface through scripting (e.g., by spawning new windows, causing dialog boxes to appear, etc.), users must be able to override changes that make the user agent or document inaccessible.

Otherwise, users who are blind, have visual impairments, some types of learning disabilities, or any user who cannot or has chosen not to view the primary representation of information will not know content of the information on the page.

The following checkpoints state which styles the user must be able to control. The checkpoints also require user agents to allow users to override author styles [p. 22] and user agent defaults [p. 22] . The checkpoints apply to alternative representations of content as well as primary content.

Checkpoints for fonts and colors:

6.1 Allow the user to control font family. [Priority 1]
6.2 Allow the user to control font size. [Priority 1]
6.3 Allow the user to control foreground color. [Priority 1]
6.4 Allow the user to control background color. [Priority 1]
6.5 Allow the user to control selection highlighting (e.g., foreground and background color). [Priority 1]
6.6 Allow the user to control focus highlighting (e.g., foreground and background color). [Priority 1]

See also checkpoints checkpoint 7.12 and checkpoint 7.13.

Checkpoints for applets and animations:

6.7 Allow the user to control animation rate. [Priority 2]

Checkpoints for video.

6.8 Allow the user to control video frame rates. [Priority 1]
6.9 Allow the user to control the position of captions. [Priority 1]
6.10 Allow the user to start, stop, pause, and rewind video. [Priority 2]

Checkpoints for audio:

6.11 Allow the user to control audio playback rate. [Priority 1]
6.12 Allow the user to control audio volume. [Priority 2]
6.13 Allow the user to start, stop, pause, and rewind audio. [Priority 2]

Checkpoints for speech:

6.14 Allow the user to control speech playback rate. [Priority 1]
6.15 Allow the user to control speech volume, pitch, gender and other articulation characteristics. [Priority 2]

Checkpoints for changes to the user interface:

6.16 When new windows or user interface components are spawned, allow the user to control window size and position. [Priority 2]

Guideline 7. Ensure user access to document content

Ensure that users have access to primary as well as author-supplied alternative representations of content (descriptions of images, captions for video or audio, etc.)

Otherwise, some users cannot perceive the primary content due to a disability or a technological limitation (e.g., browser configured not to display images).

General checkpoints:

7.1 Ensure that the user has access to document content, including alternative representations of content. [Priority 1]

Mechanisms for specifying alternative content vary according to markup language. For instance, in HTML or SMIL, the "alt" attribute specifies alternative text for many elements. In HTML, the content of the OBJECT element is used to specify alternative content, the "summary" attribute applies to tables, etc.

7.2 *For dependent user agents. [p. 8]* Ensure that the user has access to the content of an element selected by the user. [Priority 1]

For instance, allow the user to identify a table cell with the selection and provide the user with cell content and (optionally) associated header information.

7.3 *For dependent user agents. [p. 8]* Support marked-up changes in natural language of content. For unsupported languages, notify the user of the change. [Priority 2]

7.4 When no alternative text representation has been specified, indicate what type of object is present. [Priority 2]

7.5 When alternative text has been specified explicitly as empty (i.e., an empty string), render nothing. [Priority 3]

Checkpoints for audio:

7.6 If a technology allows for more than one caption or description track for audio, allow the user to choose from among tracks. [Priority 1]

7.7 Allow the user to specify that captions for audio be rendered at the same time as the audio. [Priority 1]

Checkpoints for video:

7.8 If a technology allows for more than one caption or description track (e.g., text, video of sign language, etc.) for video, allow the user to choose from among tracks. [Priority 1]

7.9 If a technology allows for more than one audio track for video, allow the user to choose from among tracks. [Priority 1]

7.10 Allow the user to specify that text descriptions of video be rendered at the same time as the video. [Priority 1]

7.11 Allow the user to specify that auditory descriptions of video be rendered at the same time as the video. [Priority 1]

Checkpoints for links:

7.12 Provide a mechanism (e.g., through style sheets) to distinguish visited links from unvisited links. [Priority 3]

Note. Using color as the only distinguishing factor does not suffice since color may not be perceivable by all users or rendered by all devices.

7.13 Allow the user to specify (e.g., through style sheets) that images used in links must have borders. [Priority 3]

Guideline 8. Provide navigation mechanisms

Subhead here.

Note. For all search and navigation functions, the user agent should follow system conventions for using selection [p. 22] and focus [p. 21] mechanisms. For instance, the selection should be used to identify the results of a text search, the focus should identify active elements [p. 21] during sequential navigation of active elements, etc.

8.1 Allow the user to navigate views (notably those with frame viewports). [Priority 1]
Note. Navigating into a view makes it the current view.
8.2 Keep track of the user's point of regard [p. 7] in each view and restore it when the user returns to the view. [Priority 1]
8.3 *For dependent user agents. [p. 8]* Allow the user to navigate among table cells of a table (notably left and right within a row and up and down within a column). [Priority 1]
8.4 Allow the user to navigate among all active elements [p. 21] in the document. [Priority 2]
Navigation mechanisms may range from sequential (e.g., tabbing navigation) to direct (e.g., by entering link text) to searching on active elements only (e.g., based on form control text, associated labels, or form control names).
8.5 Allow the user to search for rendered text content, including alternative text content. [Priority 2]
8.6 Allow the user to navigate the document structure. [Priority 2]
For example, allow the user to navigate familiar elements of a document: paragraphs, tables, headers, lists, etc.
8.7 Allow the user to configure structured navigation. [Priority 3]
For example, allow the user to navigate only paragraphs, or only headers and paragraphs, etc.

Guideline 9. Help orient the user

Subhead here.

Document information:

Users that are viewing documents through linear channels of perception like speech (since speech is temporal in nature) and tactile displays (current tactile technology is limited in the amount of information that can be displayed) have difficulty maintaining a sense of their relative position in a document. The meaning of "relative position" depends on the situation. It may mean the distance from the beginning of the document to the point of regard [p. 23] (how much of the document has been read), it may mean the cell currently being examined in a table, or the position of the current document in a set of documents.

For people with visual impairments, blindness, or certain types of learning disabilities, it is important that the point of regard [p. 23] remain as stable as possible. The user agent should not disturb the user's point of regard by shifting focus to a different frame or window when an event occurs without notifying the user of the change.

View, selection, and focus information:

9.1 Provide a mechanism for highlighting and identifying (through a standard interface where available) the current view. [Priority 1]
9.2 Provide a mechanism for highlighting and identifying (through a standard interface where available) the user selection. [Priority 1]
9.3 Provide a mechanism for highlighting and identifying (through a standard interface where available) the current focus. [Priority 1]
9.4 *For dependent user agents. [p. 8]* Provide the user with information about the number of viewports [p. 23] . [Priority 2]
9.5 Ensure that when the selection changes, it is in the viewport [p. 23] after the change. [Priority 2]
9.6 Ensure that when the focus changes, it is in the viewport [p. 23] after the change. [Priority 2]
9.7 Identify a frame selected by the user. [Priority 2]

Document information:

9.8 *For dependent user agents. [p. 8]* Allow the user to view a document outline constructed from its structural elements (e.g., from header and list elements). [Priority 2]
9.9 Make available the primary language of a document's content. [Priority 3]
For example, use HTTP header information or language attribute information from the document root.
9.10 Make available the number of links (to distinct targets) in a document. [Priority 3]
9.11 Make available the number of visited links (to distinct targets) in a document. [Priority 3]
9.12 Make available the number of tables in a document. [Priority 3]
9.13 Make available the number of form controls in a document. [Priority 3]
9.14 Make available what portion of the document has loaded. [Priority 3]

Link information:

9.15 Identify a link selected by the user. [Priority 3]
For instance, the link's integer position in the document.
9.16 Make available whether a chosen link (target) has already been visited. [Priority 3]
9.17 Make available whether a chosen link (target) is local to the document. [Priority 3]

9.18 Make available whether following a link will involve a fee. [Priority 3]

Table information:

9.19 Identify a table selected by the user. [Priority 3]
For instance, the table's integer position in the document.
9.20 Make available the dimensions of a chosen table. [Priority 3]

Table cell information:

9.21 *For dependent user agents. [p. 8]* Provide access to header information for a table cell selected by the user. [Priority 1]
9.22 Identify the table containing a table cell selected by the user. [Priority 3]
9.23 Make available the coordinates in the current table of a selected table cell. [Priority 3]

Form control information:

9.24 Provide the user with access to any label explicitly associated with a form control. [Priority 2]

Consistency:

9.25 Maintain consistent user agent behavior and default configurations between software releases. Consistency is less important than accessibility and adoption of system conventions. [Priority 3]
Changes to layout of user interface controls, behavior of existing functionalities, default keyboard configuration, and other user agent features should not be arbitrary.

Guideline 10. Notify the user of document changes

It is important to alert users, in an output device-independent fashion, when important events occur during a browsing session.

To avoid confusion that the effects of scripts may cause, users should be notified when scripts are executed (or be able to disable scripts [p. 13] entirely). This is also important for security reasons; users should be able to decide whether to allow scripts to execute on their machines.

10.1 Provide information about document changes (to the user and through programming interfaces). [Priority 1]
10.2 Allow the user to configure the user agent for notification of certain types of document changes only. [Priority 3]
10.3 Notify the user when enough of the document has loaded in order to begin to browse. [Priority 3]

10.4 Notify the user when document loading has stalled. [Priority 3]
10.5 Allow the user to request to be prompted before a form is submitted. [Priority 3]

Guideline 11. Support applicable W3C technologies and guidelines

User agents should support applicable W3C technologies and in particular the accessibility features defined for the technology.

The current guidelines recommend support for W3C technologies (e.g., HTML, CSS, MathML, SMIL, etc.) for several reasons:

- W3C technologies include "built-in" accessibility features.
- W3C specifications undergo early review to ensure that accessibility issues are considered during the design phase.
- W3C specifications are developed in an open, industry consensus process.

11.1 Implement the accessibility features defined for supported technologies. [Priority 1]
Note. The Techniques Document ([UA-TECHNIQUES] [p. 25]) discusses accessibility features of W3C technologies.
11.2 Support appropriate W3C Recommendations. [Priority 2]
For instance, for document markup, support HTML and XML; for style sheets, support CSS; for mathematics, support MathML; for multimedia, support SMIL, etc.

Guideline 12. Observe system conventions and standard interfaces

When a user agent communicates with other software (dependent user agents, the operating system, plug-ins), must do so through applicable interfaces. To promote interoperability, open standards should be used wherever possible.

Some operating systems have operating system-level flags and settings that are pertinent to accessibility, such as high-contrast colors and "show" sounds for people with hearing impairments. User agents should take these global settings into account for their own settings.

12.1 Use and provide accessible interfaces to other technologies. [Priority 1]
To promote interoperability, open standards and W3C specifications should be used wherever possible.
12.2 Provide programmatic read and write access to user agent functionalities and user interface controls (including selection and focus) by using operating system and development language accessibility resources and conventions. [Priority 1]

12.3 Notify dependent user agents of changes to the document and user interface controls (including selection and focus) by using operating system and development language accessibility resources and conventions. [Priority 1]
12.4 *For desktop graphical browsers. [p. 8]* Comply with W3C Document Object Model specifications and export interfaces defined by those specifications. [Priority 1]

For example, refer to [DOM, Level 1 [p. 25]]. **Note.** The DOM Level 1 specification states that "DOM applications may provide additional interfaces and objects not found in this specification and still be considered DOM compliant."

12.5 *For desktop graphical browsers. [p. 8]* Provide programmatic exchange of information in a timely manner. [Priority 2]

This is important for synchronous alternative renderings and simulation of events.

12.6 Follow operating system conventions and accessibility settings. In particular, follow conventions for user interface design, default keyboard configuration, product installation, and documentation. [Priority 2]

Refer also to checkpoint 2.7.

Appendix A. -- Glossary

Applicable checkpoint

A checkpoint applies to a user agent unless:

- The checkpoint definition states explicitly that it only applies to a different class of user agent.
- The checkpoint addresses a content type (script, image, video, sound, applets, etc.) that the user agent does not recognize. [p. 22]
- The checkpoint refers to a content type that the user agent recognizes [p. 22] but does not support natively [p. 21] .
- The checkpoint refers to the properties of an embedded object (e.g., video or animation rate) that may not be controlled or accessed by the user agent.

Documents, Elements, and Attributes

A document may be seen as a hierarchy of *elements.* Elements are defined by a language specification (e.g., HTML 4.0 or an XML application). Each element may have content, which generally contributes to the document's content. Elements may also have *attributes* that take values. An element's *rendered content* is that which a user agent renders for the element. This may be what lies between the element's start and end tags, the value of an attribute (c.f. the "alt", "title", and "longdesc" attributes in HTML), or external data (e.g., the IMG element in HTML). Rendering is not limited to visual displays alone, but also includes audio (speech and sound) and tactile displays (braille and haptic displays).

Since rendered content is not always accessible, authors must specify *alternative representations of content* that user agents must make available to

users or software that require it (in place of and/or in addition to the "primary" content). Alternative representations may take a variety of forms including alternative text, text captions, and auditory descriptions. The Techniques Document ([UA-TECHNIQUES] [p. 25]) describes the different mechanisms authors use to supply alternative representations of content. Please also consult the Web Content Accessibility Guidelines ([WAI-WEBCONTENT] [p. 26]).

Events and scripting

When certain *events* occur (document loading or unloading events, mouse press or hover events, keyboard events, etc.), user agents often perform some task (e.g., execute a script). For instance, in most user agents, when a mouse button is released over a link, the link is activated and the linked resource retrieved. User agents may also execute author-defined scripts when certain events occur. The script bound to a particular event is called an *event handler.*
Note. The interaction of HTML, style sheets, the Document Object Model [DOM1] [p. 25] and scripting is commonly referred to as "Dynamic HTML" or DHTML. However, as there is no W3C specification that formally defines DHTML, this document will only refer to event handlers and scripts.

Focus

The user focus designates an *active element* in a document. Which elements are active depends on the document language and whether the features are supported by the user agent. In HTML documents, for example, active elements include links, form controls, elements with a value for the "longdesc" attribute, and elements with associated scripts (event handlers). An element with the focus may be *activated* through any number of mechanisms, including the mouse, keyboard, an API, etc.

The effect of activation depends on the element. For instance, when a link is activated, the user agent generally retrieves the linked resource, which may be another Web page, program, etc. When a form control is activated, it may change state (e.g., check boxes) or may take user input (e.g., a text field). Activating an element with a script assigned for that particular activation mechanism (e.g., mouse down event, key press event, etc.) causes the script to be executed.

A view has at most one focus. When several views co-exist, each may have a focus, but only one is active, called the *current focus.* The current focus is generally presented (e.g., highlighted) in a way that makes it stand out.

Insertion point

The insertion point is the location where document editing takes place. The insertion point may be set by the user (e.g., by a pointing device or the keyboard editing keys) or through an application programming interface (API). A view has at most one insertion point. When several views co-exist, each may have an insertion point, but only one is active, called the *current insertion point*

The insertion point is generally rendered specially (e.g., on the screen, by a vertical bar or similar cursor).

Native support

A user agent supports a feature natively if it does not require another piece of software (e.g., plug-in or external program) for support. Native support does not

preclude more extensive support for accessibility by assistive technologies, so user agents must still make information available through programming interfaces.

Properties, Values, and Defaults

A user agent renders a document by applying formatting algorithms and style information to the document's elements. Formatting depends on a number of factors, including where the document is rendered: on screen, paper, through speakers, a braille device, a mobile device, etc. Style information (e.g., fonts, colors, voice inflection, etc.) may come from the elements themselves (e.g., certain style attributes in HTML), from style sheets, or from user agent settings. For the purposes of these guidelines, each formatting or style option is governed by a *property* and each property may take one value from a set of legal values. (The term "property" in this document has the meaning ascribed in the CSS2 Recommendation [p. 25] .) A reference to "styles" in this document means a set of style-related properties.

The value given to a property by a user agent when it is started up is called the property's *default value.* User agents may allow users to change default values through a variety of mechanisms (e.g., the user interface, style sheets, initialization files, etc.).

Once the user agent is running, the value of a property for a given document or part of a document may be changed from the default value. The value of the property at a given moment is called its *current value.* Note that changes in the current value of a property do not change its default value.

Current values may come from documents, style sheets, scripts, or the user interface. Values that come from documents, their associated style sheets, or via a server are called *author styles.* Values that come from user interface settings, user style sheets, or other user interactions are called *user styles.*

Recognize

A user agent is said to recognize markup, content types, or rendering effects when it can identify (through built-in mechanisms, DTDs, style sheets, headers, etc) the information. For instance, HTML 3.2 user agents may not recognize the new elements or attributes of HTML 4.0. Similarly, a user agent may recognize visually blinking content specified by elements or attributes, but may not recognize that an applet is blinking. The Techniques Document ([UA-TECHNIQUES] [p. 25]) discusses some content that affects accessibility and should be recognized as such.

Selection

The user selection generally specifies a range of content (text, images, etc.) in a document. The range may be restricted to the content of a single element or may span several elements. The selection may be used for a variety of purposes: for cut and paste operations, to designate a specific element in a document, to identify what a screen reader should read, etc.

The user selection may be set by the user (e.g., by a pointing device or the keyboard) or through an application programming interface (API). A view has at most one user selection. When several views co-exist, each may have a user selection, but only one is active, called the *current user selection.*

The user selection is usually presented in a way the stands out (e.g., highlighted). On the screen, the selection may be highlighted using colors, fonts, graphics, or other mechanisms. Highlighted text is often used by dependent user agents [p. 4] to indicate through speech or braille output what the user wants to read. Most screen readers are sensitive to highlight colors. Dependent user agents may provide alternative presentation of the selection through speech, enlargement, or refreshable braille display.

Both the current focus and the current user selection must be in the same view, called the *current view.* The current view is generally highlighted when several views co-exist.

Views, Viewports, and Point of Regard

User agents may handle different types of source information: documents, sound objects, video objects, etc. The user perceives the information through a *viewport*, which may be a window, frame, a piece of paper, a speaker, a virtual magnifying glass, etc. A viewport may contain another viewport (e.g., nested frames, plug-ins, etc.).

User agents may render the same source information in a variety of ways; each rendering is called a *view.* For instance, a user agent may allow users to view a document in one window and a generated list of headers for the document in another.

The view is *how* source information is rendered and the viewport is *where* it is rendered.

Generally, viewports give users access to all rendered information, though not always at once. For example, a video player shows a certain number of frames per second, but allows the user to rewind and fast forward. A visual browser viewport generally features scrollbars or some other paging mechanism that allows the user to bring the rendered content into the viewport.

The content currently available in the viewport is called the user's *point of regard*. The point of regard may be a two dimensional area (e.g., for graphical rendering) or a single point (e.g., for aural rendering or voice browsing). User agents should not change the point of regard unexpectedly as this can disorient users.

---

5. Acknowledgments

Many thanks to the following people who have contributed through review and comment: Paul Adelson, James Allan, Denis Anson, Kitch Barnicle, Harvey Bingham, Olivier Borius, Judy Brewer, Bryan Campbell, Kevin Carey, Wendy Chisholm, David Clark, Chetz Colwell, Wilson Craig, Nir Dagan, Daniel Dardailler, B. K. Delong, Neal Ewers, Geoff Freed, John Gardner, Al Gilman, Larry Goldberg, John Grotting, Markku Hakkinen, Earle Harrison, Chris Hasser, Kathy Hewitt, Philipp Hoschka, Masayasu Ishikawa, Phill Jenkins, Jan Kärrman (for help with html2ps), Leonard Kasday, George Kerscher, Marja-Riitta Koivunen, Josh Krieger, Catherine Laws, Greg Lowney, Scott Luebking, William Loughborough, Napoleon Maou, Charles McCathieNevile, Masafumi Nakane, Mark Novak, Charles Oppermann, Mike Paciello, David Pawson, Michael Pederson, Helen Petrie, David Poehlman, Michael Pieper, Jan Richards, Hans Riesebos, Joe Roeder, Lakespur L. Roca, Greg Rosmaita, Lloyd Rutledge, Liam Quinn, T.V. Raman, Robert Savellis, Rich Schwerdtfeger, Constantine Stephanidis, Jim Thatcher, Jutta Treviranus, Claus Thogersen, Steve Tyler, Gregg Vanderheiden, Jaap van Lelieveld, Jon S. von Tetzchner, Willie Walker, Ben Weiss, Evan Wies, Chris Wilson, Henk Wittingen, and Tom Wlodkowski,

6. References

[CSS1]
"CSS, level 1 Recommendation", B. Bos, H. Wium Lie, eds. The CSS1 Recommendation is:
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:
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:
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:
http://www.w3.org/TR/1998/REC-html40-19980424.

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

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

[RFC2119]
"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt

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

[UA-TECHNIQUES]
"Techniques for User Agent Accessibility Guidelines 1.0", J. Gunderson, I. Jacobs, eds. This document explains how to implement the checkpoints defined in "User Agent Accessibility Guidelines 1.0". The latest draft of the techniques is available at: http://www.w3.org/WAI/UA/WAI-USERAGENT-TECHS/

[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-WEBCONTENT]
"Web Content Accessibility Guidelines", W. Chisholm, G. Vanderheiden, and I. Jacobs, eds., 5 May 1999. This W3C Recommendation is http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990505.

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