User Agent Accessibility Guidelines 1.0

W3C Working Draft 21 August 2002

This version:
http://www.w3.org/TR/2002/WD-UAAG10-20020821/
Latest version:
http://www.w3.org/TR/UAAG10/
Previous version:
http://www.w3.org/TR/2001/CR-UAAG10-20010912/

Editors:
Ian Jacobs, W3C
Jon Gunderson, University of Illinois at Urbana-Champaign
Eric Hansen, Educational Testing Service
Authors and Contributors:
See acknowledgements .

This document is also available in these non-normative formats: single HTML, plain text, gzip PostScript, Black/white gzip PostScript, gzip PDF, gzip tar file of HTML, and zip archive of HTML. **Note:** Some user agents unzip the gzipped files on the fly without changing the file suffix. If you encounter problems reading the gzipped files, remove the `.gz` suffix and try again.

Abstract

This document provides guidelines for designing user agents that lower barriers to Web accessibility for people with disabilities (visual, hearing, physical, cognitive, and neurological). User agents include HTML browsers and other types of software that retrieve and render Web content . A user agent that conforms to these guidelines will promote accessibility through its own user interface and through other internal facilities, including its ability to communicate with other technologies (especially assistive technologies). Furthermore, all users, not just users with disabilities, are expected to find conforming user agents to be more usable.

In addition to helping developers of HTML browsers, media players, etc., this document will also benefit developers of assistive technologies because it explains what types of information and control an assistive technology may expect from a conforming user agent. Technologies not addressed directly by this document (e.g.,

technologies for braille rendering) will be essential to ensuring Web access for some users with disabilities.

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the W3C.

This is the 21 August 2002 Last Call Working Draft of "User Agent Accessibility Guidelines 1.0". The last call review period ends 18 September 2002. Last Call Working Draft status is described in section 5.2.2 of the Process Document. Since the previous Candidate Recommendation draft, the UAWG has gathered implementation experience and clarified the document based on in-depth discussions with user agent and assistive technology developers.

As a result of the second Candidate Recommendation period, the UAWG:

1. deleted or modified some checkpoint provisions with low implementation experience;
2. retained some checkpoints despite low implementation experience;
3. added one checkpoint regarding API access to some rendering information.

The chapter on conformance has also been greatly simplified since the Candidate Recommendation. The complete list of changes is available on the Web.

In this review, the User Agent Accessibility Guidelines Working Group is primarily interested in comments on changes since the Candidate Recommendation. The Working Group does not expect to make substantial changes to this document as the result of this review; the document has already received substantial technical review. The UAWG does expect to make clarifications and to record issues to be addressed after UAAG 1.0.

The latest information regarding patent disclosures related to this document is available on the Web. As of this publication, there are no disclosures.

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

Please send comments about this document to the public mailing list w3c-wai-ua@w3.org; public archives are available.

This document is part of a series of accessibility documents published by the Web Accessibility Initiative (WAI) of the World Wide Web Consortium (W3C). WAI Accessibility Guidelines are produced as part of the WAI Technical Activity. The goals of the User Agent Accessibility Guidelines Working Group are described in the charter.

A list of current W3C Recommendations and other technical documents can be found at the W3C Web site.

Table of contents

1. Introduction

This document specifies requirements that, if satisfied by user agent developers, will lower barriers to accessibility. This document includes:

- This introduction, which provides context for understanding the requirements listed in section 2 .
- Section 2 explains twelve general principles of accessible design, called "guidelines". Each guideline consists of a list of requirements, called "checkpoints", which must be satisfied in order to conform to this document.
- Section 3 explains how to make claims that software components satisfy the requirements of section 2.
- An appendix offers a summary of this document's principal goals and structure *[UAAG10-SUMMARY]* .
- A second appendix lists all the checkpoints for convenient reference (e.g., as a tool for developers to evaluate software for conformance) *[UAAG10-CHECKLIST]* .

A separate document, entitled "Techniques for User Agent Accessibility Guidelines 1.0" (the "Techniques document" from here on) *[UAAG10-TECHS]* , provides suggestions and examples of how each checkpoint might be satisfied. It also includes references to other accessibility resources (such as platform-specific software accessibility guidelines) that provide additional information on how a user agent may satisfy each checkpoint. The techniques in the Techniques document are informative examples only, and other strategies may be used or required to satisfy the checkpoints. The Techniques document is expected to be updated more frequently than the current guidelines. Developers, W3C Working Groups, users, and others are encouraged to contribute techniques.

1.1 Relation to WAI accessibility guidelines

"User Agent Accessibility Guidelines 1.0" (UAAG 1.0) is part of a series of accessibility guidelines published by the Web Accessibility Initiative (WAI). The documents in this series reflect an accessibility model in which Web content authors, format designers, and software developers have roles in ensuring that users with disabilities have access to the Web. The accessibility-related interests of these stakeholders intersect and complement each other as follows:

- Designers of formats (e.g., HTML, XHTML, XML, SVG, SMIL, MathML, etc.) and protocols (e.g., HTTP) create specifications that allow communication on the Web. Format designers include features in these specifications that authors should use to create accessible content and that user agents should support through an accessible user interface. The "XML Accessibility Guidelines (XAG)" *[XAG10]* explains the responsibilities of XML format designers; many XAG requirements make sense for non-XML formats as well.
- Authors make use of the accessibility features of different format specifications, use markup appropriately, write in clear and simple language, organize a Web

site consistently, etc. The "Web Content Accessibility Guidelines 1.0" *[WCAG10]* explains the responsibilities of authors in meeting the needs of users with disabilities. The "Web Content Accessibility Guidelines (WCAG) 1.0" is considered the reference for what defines accessible Web content. The "Authoring Tool Accessibility Guidelines 1.0" *[ATAG10]* explains the responsibilities of authoring tool developers. An accessible authoring tool facilitates the creation of accessible Web content and may be operated by users with disabilities.
- User agent developers design software that meets the needs of users with disabilities through conformance to other specifications, an accessible user interface, accessible documentation, and communication with other software (notably assistive technologies).

The requirements of this document interact with those of the "Web Content Accessibility Guidelines 1.0" *[WCAG10]* in a number of ways:

- UAAG 1.0 checkpoint 8.1 requires implementation of the accessibility features of specifications. Features are those identified as such and those that satisfy all of the requirements of WCAG 1.0 *[WCAG10]* .
- UAAG 1.0 checkpoint 12.1 requires conformance to WCAG 1.0 for user agent documentation.
- UAAG 1.0 also incorporates some terms and concepts from WCAG 1.0, a consequence of fact that the documents were designed to complement one another.

Some requirements of this document take into account limitations of formats, authors, and designers. Formats generally do not enable authors to encode all of their knowledge in a way that a user agent can recognize 100%. A format may lack features required for accessibility. An author may not make use of the accessibility features of a format or may misuse a format (which can cause problems for user agents). A user agent designer may not implement a format specification correctly or completely.

Some of these limitations are taken into account as follows:

- UAAG 1.0 includes requirements to satisfy the expectations set by WCAG 1.0 "until user agent" clauses. These clauses make additional requirements of authors in order to compensate for some limitations of deployed user agents.
- UAAG 1.0 includes several repair requirements (e.g., checkpoints checkpoint 2.7 and checkpoint 2.10) for cases where content does not conform to WCAG 1.0. Furthermore, this document includes some requirements to address certain widespread authoring practices that are discouraged because they may cause accessibility or usability problems (e.g., some uses of HTML frames).
- Except for the indicated repair checkpoints, UAAG 1.0 only requires user agents to handle what may be recognized through protocols and formats. For example, user agents are not expected to recognize that the author has used "clear and simple" language to express ideas (WCAG 1.0, checkpoint 14.1). See the

section on checkpoint applicability for more information about what the user agent is expected to recognize.

The Web Accessibility Initiative provides other resources and educational materials to promote Web accessibility. Resources include information about accessibility policies, links to translations of WAI materials into languages other than English, information about specialized user agents and other tools, accessibility training resources, and more.

Note: The Web Accessibility Initiative is developing new versions of both the Web Content Accessibility Guidelines and the Authoring Tool Accessibility Guidelines. UAAG 1.0 refers only to the WCAG 1.0 and ATAG 1.0 Recommendations, which will remain available and unchanged.

1.2 Target user agents

This document was designed specifically to improve the accessibility of user agents with multimedia capabilities running in the following type of environment (typically that of a desktop computer):

- The operating environment includes a keyboard (or keyboard equivalent);
- Assistive technologies may be used in the operating environment and may communicate with the conforming user agent;

The target user agent is one designed for the general public to handle general-purpose content in ordinary operating conditions.

This document is not designed so that user agents on other types of platforms (e.g., handheld devices, kiosks, etc.) will readily conform. This document does not *forbid* conformance by any user agent, but some requirements (e.g., implementation of certain application programming interfaces, or APIs) are not likely to be satisfied on environments other than the target environment. Future work by the UAWG may address the accessibility of user agents running on handheld devices, etc.

Technologies not addressed directly by this document (e.g., those for braille rendering) will be essential to ensuring Web access for some users with disabilities. Note that the ability of conforming user agents to communicate well with assistive technologies will depend in part on the willingness of assistive technology developers to follow the same standards and conventions for communication.

1.2.1 Composition of conforming user agents

In general, a conforming user agent will consist of several coordinated components, such as a Web browser, a multimedia player, several plug-ins, features or applications provided by the operating environment, and documentation distributed with the software or available on the Web. These components may run on the user's computer or on a server. A conforming user agent may also include assistive technologies and applications provided by the operating environment. The current document places no restrictions on the type or number of components used for conformance.

This does not mean that every component that one has chosen as part of the user agent has to satisfy every single requirement; some requirements may not be relevant for a particular component. For instance, if a component does not have a user interface, it would not be required to satisfy the user interface requirements. On the other hand, if a component has a user interface, that user interface would be subject to the requirements of this document. Conformance addresses the composite user agent *as a whole*.

1.2.2 Use of operating environment features

To satisfy the requirements of this document, developers are encouraged to adopt operating environment conventions and features that benefit accessibility. When an operating environment feature (e.g., the operating system's audio control panel, including its user interface) is adopted to satisfy the requirements of this document, it is part of the user agent.

See additional information on conformance of user agents running in multiple operating environments .

1.3 Known limitations of this document

People with (or without) disabilities access the Web with widely varying sets of capabilities, software, and hardware. Some users with disabilities:

- May not be able to see, hear, move, or speak.
- May not be able to perceive, read, or process some types of information easily or at all.
- May not have or be able to use a keyboard or pointing device.

This document does not include requirements to meet all known accessibility needs. Some known limitations of this document include the following:

Input modalities
: This document only includes requirements for keyboard, pointing device, and voice input modalities. This document includes several checkpoints related to voice input as part of general input requirements (e.g., the checkpoints of guideline 7 and guideline 11) but does not otherwise address voice-based navigation or control.
Note: The UAWG intends to coordinate further work on the topics of voice input and synthesized speech rendering with groups in W3C's Voice Browser Activity.

Output modalities
: This document does not include requirements for braille rendering. Some requirements are specific to graphical rendering and others specific to audio output or synthesized speech output. Speech rendering requirements are made by checkpoint 4.9 to checkpoint 4.13. Many of the requirements of this document are generic enough to apply to a variety of output modalities, including braille. User agents conform to this document by supporting some

combination of graphical and audio/speech rendering output; see the section on Content type labels for more information.

Size and color of non-text content
: This document includes some checkpoints to ensure that the user is able to control the size and color of visually rendered text content (checkpoints 4.1 and 4.3). This document does not in general address control of the size and color of visually rendered non-text content .
: **Note:** Resizing capabilities may be required for conformance to other specifications (e.g., Scalable Vector Graphics *[SVG]*).

Background image interference
: The requirement of checkpoint 3.1 to allow the user to turn off rendering of background images does not extend to multi-layered rendering.

User control of every user interface component
: This document distinguishes user interface features that are part of the user agent user interface and those that are part of content . Some checkpoints (e.g., those in guideline 5) require user control over rendering and behavior that is driven by content only. This document does not always explicitly require the same control over features of the user agent user interface. Nevertheless, this document (see checkpoint 7.3) does require user agents to follow software usability guidelines, which should include requirements for user control over user interface behavior.
: **Note:** It is more difficult for users to distinguish content from user interface when both are rendered as sound in one temporal dimension , than it is when both are rendered visually in two spatial dimensions. Thus, developers of user agents that include audio output or synthesized speech output are encouraged to apply the requirements of this document to both content and user agent components.

Time parameters
: This document includes requirements for control of some time parameters (including checkpoint 2.4, checkpoint 4.4, checkpoint 4.5, and checkpoint 4.9). The requirements are for time parameters that the user agent recognizes and controls. This document does not include requirements for control of time parameters managed on the server.

Digital rights management
: The User Agent Accessibility Guidelines Working Group recognizes that further work is necessary in the area of digital rights management as it relates to accessibility. Digital rights management refers to methods of describing and perhaps enforcing intellectual property associated with Web resources.

Note: The User Agent Accessibility Guidelines Working Group may address these topics in a future version of the User Agent Accessibility Guidelines. Even though UAAG 1.0 does not address these topics, user agent developers are encouraged to consider them in their designs.

1.4 Relation to general software design guidelines and other specifications

One the goals of the authors of this document is to ensure that the requirements are compatible with other good software design practices. However, this document does not purport to be a complete guide to good software design. For instance, the general topic of user interface design for computer software exceeds the scope of this document, though some user interface requirements have been included because of their importance to accessibility. The Techniques document *[UAAG10-TECHS]* includes some references to general software design guidelines and platform-specific accessibility guidelines (see checkpoint 7.3). Involving people with disabilities in the design and testing of software will generally improve the accessibility of the software.

This document promotes conformance to other specifications as part of accessible design. Conformance to specifications makes it easier to design assistive technologies, and helps ensure that built-in accessibility functions are implemented.

This document also includes some requirements to implement an accessibility feature that may only be optional in another specification.

In rare cases, a requirement in UAAG 1.0 may conflict with a requirement in another specification. UAAG 1.0 does not include requirements for resolving this conflict, but the authors of this document anticipate that developers will consider accessibility implications in determining how to resolve the conflict.

1.4.1 Installation

Installation is an important aspect of both accessibility and general software usability. On platforms where a user can install a user agent, the installation (and update) procedures need to be accessible. Furthermore, the installation procedure should provide and install all components necessary to satisfy the requirements of this document, as the risk of installation failure increases with the number of components (e.g., plug-ins) to be installed.

This document does not include a checkpoint requiring that installation procedures be accessible. Since this document considers installation to be part of software usage, the different aspects of installation (user interface, documentation, operating environment conventions, etc.) are already covered by the complete set of checkpoints.

1.5 Security considerations

Some of the requirements of this document have security implications: communication through APIs, allowing programmatic read and write access to content and user interface control , etc. This document assumes that features required by this document will be built on top of an underlying security architecture. Consequently, unless permitted explicitly in a checkpoint (as in checkpoint 6.5), this document grants no conformance exemptions based on security issues.

Developers should design user agents that enable communication with trusted assistive technologies. Sensitive information that the user agent can access through the user agent's user interface should also be available to assistive technologies through secure means. For instance, if the user types a password in the user agent user interface, do not communicate substitute characters (such as asterisks) through an API, but rather the real password, properly encrypted.

Note also that appropriate user agent behavior with respect to security may depend on the user's context. For instance, hiding typed passwords with asterisks is much less important for someone alone in a room than for someone in a crowded room. Similarly, while unencrypted passwords rendered as synthesized speech should not be broadcast in a crowded room, they may pose no security risk if the user is wearing an earphone.

For information related to security, refer to "XML-Signature Syntax and Processing" *[XMLDSIG]* and "XML Encryption Syntax and Processing" *[XMLENC]* .

1.6 User control

This document emphasizes the goal of ensuring that users, including users with disabilities, have control over their environment for accessing the Web. Key methods for achieving that goal include: optional self-pacing, configurability, device-independence, interoperability, direct support for both graphical and auditory output, and adherence to published conventions. Chapter 2 addresses these issues in detail.

This document also acknowledges the importance of author preferences and the proper implementation of specifications. However, this document includes requirements to override certain author preferences when the user would not otherwise be able to access that content.

1.6.1 Control of automatic behavior

Many of the requirements in this document give the user additional control over behavior that would otherwise occur automatically. For instance, there is a requirement to allow configuration to not open a viewport automatically (checkpoint 5.3) and one that requires user confirmation before submitting a form (checkpoint 5.5). This type of manual configuration option may be essential for some users with disabilities, since automatic behavior may be disorienting or interfere with navigation.

1.6.2 Configurability

This document includes requirements for users with a variety of disabilities, in part because some users may have more than one disability. In some cases, it may appear that two requirements contradict each other. For instance, a user with a physical disability may prefer that the user agent offer more automatic behavior (to reduce demand for physical effort) than a user with a cognitive disability (for whom automatic behavior may cause confusion). Thus, many of the requirements in this document involve configuration as one way to ensure that a functionality designed to improve accessibility for one user does not interfere with accessibility for another. Also, since a default user agent setting may be useful for one user but interfere with accessibility for another, this document prefers configuration requirements to requirements for default settings. Finally, there may be some cases where, for some content, a feature required by this document is ineffective or causes content to be less accessible, making it imperative that the user be able to turn off the feature.

To avoid the risk that users are overwhelmed by an abundance of configuration options, this document includes requirements that promote ease of configuration and documentation of accessibility features (see guideline 12).

1.6.3 Device independence, spatial independence, and temporal independence

Many requirements in this document promote different kinds of independence:

- Input and output device independence. This document includes some requirements to promote device-independence natively, as well as requirements for interoperability with assistive technologies that provide complementary input and output functionalities.
- Spatial independence. Some users may not navigate effectively in two-dimensional visual space (e.g., users who do not use a pointing device) or may be constrained to one temporal dimension (e.g., users of audio-only output).
- Temporal independence. Some users (e.g., users with a physical or cognitive disability) may not be able to interact with content that changes over time, or interaction with content that is time-sensitive.

1.6.4 Additional benefits of accessible user agent design

In meeting the goals of users with disabilities, user agent developers will also to improve access to the Web for users in general. For example, users without disabilities:

- may have a text-only screen, a small screen, or a slow Internet connection (e.g., via a mobile phone browser). These users are likely to benefit from the same features that provide access to people with low vision or blindness.
- may be in a situation where their eyes, ears, or hands are busy or interfered with (e.g., driving to work, working in a noisy environment, etc.). These users are likely to benefit from the same features that provide access to people who cannot use a mouse or keyboard due to a visual, hearing, or physical disability.
- may not understand fluently the natural language of spoken content. These users are likely to benefit from the same visual rendering of text equivalents that make spoken language accessible to people with a hearing disability.

Software that satisfies the requirements of this document is expected to be more flexible, manageable, extensible, and beneficial to all users. For example, a user agent architecture that allows programmatic access to content and the user interface will encourage software modularity and reuse, and will enable operation by scripting tools and automated test engines in addition to assistive technologies.

2. The user agent accessibility guidelines

The twelve guidelines in this document state general principles for the development of accessible user agents. Each guideline includes:

- The guideline number.
- The guideline title.
- The rationale behind the guideline and identification of some groups of users who benefit from it.
- A list of checkpoint definitions. This list may be split into groups of related checkpoints. For instance, the list might be split into one group of "checkpoints for visually rendered text" and second group of "checkpoints for audio volume control"." Within each group, checkpoints are ordered according to their priority , e.g., Priority 1 before Priority 2. Within a guideline, checkpoint groupings and checkpoint order have no bearing on conformance .

Each checkpoint definition includes the following parts. Some parts are normative (i.e., relate to conformance); others are informative only.

- The checkpoint number.
- The checkpoint title. This title is not a requirement, but rather is a phrase to help readers remember an important requirement made by the checkpoint provision(s). (Informative)
- The priority of the checkpoint. (Normative)
- A link to the Techniques document *[UAAG10-TECHS]* for more information about the checkpoint: rationale, who benefits, example techniques, references, and more. (Informative)
- A list of one or more **checkpoint provisions**, which embody the requirements of the checkpoint. These requirements must be satisfied by the user agent for conformance . (Normative)
- Techniques that are sufficient for satisfying all or part of a checkpoint. (Normative when present)
- Normative inclusions and exclusions. These are qualifications about what is required (inclusion) or is not required (exclusion) to satisfy the checkpoint. Some of the inclusions are reminders about what may be required for conformance :
    1. When it might be ambiguous whether a checkpoint makes requirements for content only, the user agent user interface only, or both together, a label will state the intended scope. See the section on requirements for content, user agent features, or both for more information.
    2. When a checkpoint may be excluded from a conformance profile, it is identified by a conformance profile label . See the section on conformance profiles for more information on how a user agent may conform to this document even though it does not satisfy every checkpoint.

  (Normative when present)
- Notes about the checkpoint (beginning with the word "**Note**"). The notes clarify the scope of the checkpoint through further description, examples, cross

references, and commentary. (Informative when present)

First-time readers of the document are encouraged to read the full context provided for each checkpoint, including the guideline prose, the surrounding checkpoints (since nearby checkpoints are generally related), notes after checkpoints, and associated techniques (in the Techniques document *[UAAG10-TECHS]*). The checklist *[UAAG10-CHECKLIST]* is also a useful tool (e.g., for evaluating a user agent for conformance), but does not provide the same contextual support.

About the checkpoints

The checkpoints in this document are not generally technology-specific. They have been designed to be largely technology-independent in order to make sense for a variety of existing and future technologies. The Techniques document [UAAG10-TECHS] is an important resource to help developers understand how to apply the requirements to HTML, CSS, SMIL, and SVG, and several operating environments.

Each checkpoint is a "minimal" requirement that must be satisfied for conformance . Developers can always implement features beyond those required by this document. In some cases, it may be easier (or just better design) to implement a general feature rather than one that meets only the narrow requirement of a single checkpoint. For example, a navigable structure view of content that allows users to query elements for their properties is likely to benefit all users and may be used to satisfy a number of requirements of this document.

Some requirements have a wider impact than others. For instance, the keyboard requirements of checkpoint 1.1 have an impact on all other requirements in the document related to user input: any requirement that involves user input must be satisfied through the keyboard. Because the keyboard requirements of checkpoint 1.1 have been factored out, the other checkpoints are shorter; they are written "Allow configuration" instead of "Allow configuration so that, through the keyboard, ..."

Priorities

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

Priority 1 (P1)
: If the user agent does not satisfy this checkpoint, one or more groups of users with disabilities will find it impossible to access the Web. Satisfying this checkpoint is a basic requirement for enabling some people to access the Web.

Priority 2 (P2)
: If the user agent does not satisfy this checkpoint, one or more groups of users with disabilities will find it difficult to access the Web. Satisfying this checkpoint will remove significant barriers to Web access for some people.

Priority 3 (P3)
: If the user agent satisfies this checkpoint, one or more groups of users with disabilities will find it easier to access the Web.

This document uses the priorities as one mechanism for allowing conformance to well-defined sets of checkpoints. See the section on conformance levels for more information.

Guideline 1. Support input and output device-independence.

Ensure that the user can interact with the user agent (and the content it renders) through different input and output devices.

Since people use a variety of devices for input and output, user agent developers need to ensure redundancy in the user interface . The user may have to operate the user interface with a variety of input devices (keyboard, pointing device, voice input, etc.) and output modalities (e.g., graphical , speech, or braille rendering).

Though it may seem contradictory, enabling full user agent operation through the keyboard is an important part of promoting device-independence given today's user agents. In addition to the fact that some form of keyboard is supported in most operating environments, there are several reasons for this:

- For some users (e.g., users with blindness or physical disabilities), operating a user agent with a pointing device may be difficult or impossible since it requires tracking the pointing device position in a two-dimensional visual space. Keyboard operation generally makes fewer perceptual/motor demands for moving the pointing device to a visual target.
- Some assistive technologies that support a diversity of input and output mechanisms use keyboard APIs for communication with some user agents; see checkpoint 6.7. People who cannot or do not use a pointing device may interact with the user interface with the keyboard, through voice input, a head wand, touch screen, or other device.

While this document only requires keyboard operation for conformance , it promotes input device independence by also allowing people to claim conformance for full pointing device support or full voice support.

As a way to promote output device independence, this guideline requires support for text messages in the user interface because text may be rendered either visually, as synthesized speech, or as braille.

The API requirements of guideline 6 also promote device independence by ensuring communication with other software, including assistive technologies .

Checkpoint definitions

1.1 Full keyboard access. (P1)

1. Ensure that the user can operate through keyboard input alone any user agent functionality available through the user interface .

Normative inclusions and exclusions

1. This checkpoint excludes the requirements of checkpoint 1.3.
2. Conformance detail: For both content and user agent.

Note: For example, ensure that the user can interact with enabled elements , select content, navigate viewports, configure the user agent, access documentation, install the user agent, operate user interface controls , etc., all entirely through keyboard input.

User agents generally support at least three types of keyboard operation:

1. Direct (e.g., keyboard shortcuts such a "F1" to open the help menu; see checkpoint 11.4 for single-key access requirements),
2. Sequential (e.g., navigation through cascading menus), and
3. Spatial (e.g., when the keyboard is used to move the pointing device in two-dimensional visual space to manipulate a bitmap image).

User agents should support direct or sequential keyboard operation for all functionalities. Furthermore, the user agent should satisfy this checkpoint by offering a combination of keyboard-operable user interface controls (e.g., keyboard operable print menus and settings) and direct keyboard shortcuts (e.g., to print the current page).

It is also possible to claim conformance to this document for full support through pointing device input and/or voice input. See the section on Input modality labels .

1.2 Activate event handlers. (P1)

1. Allow the user to activate , through keyboard input alone, all event handlers that are explicitly associated with the element designated by the content focus .
2. In order to satisfy provision one of this checkpoint, the user must be able to activate as a group all event handlers of the same input device event type.

Normative inclusions and exclusions

1. Provision one of this checkpoint applies to handlers of *any* input device event type, including event types for keyboard, pointing device, and voice input.
2. The user agent is not required to allow activation of event handlers associated with a given device (e.g., the pointing device) in any order other than what the device itself allows (e.g., a mouse down event followed by a mouse drag event followed by a mouse up event).

3. The requirements for this checkpoint refer to **any** explicitly associated input device event handlers associated with an element, independent of the input modalities for which the user agent conforms. For example, suppose that an element has an explicitly associated handler for pointing device events. Even when the user agent only conforms for keyboard input (and does not conform for the pointing device, for example), this checkpoint requires the user agent to allow the user to activate that handler with the keyboard.
4. This checkpoint is mutually exclusive of checkpoint 1.1 since it may be excluded from a conformance profile , unlike other keyboard operation requirements.
5. Conformance profile labels : Events .

Note: Refer to the checkpoints of guideline 9 for more information about focus requirements.

1.3 Provide text messages. (P1)

1. Ensure that every message (e.g., prompt , alert , notification, etc.) that is a non-text element and is part of the user agent user interface has a text equivalent .

Note: For example, if the user is alerted of an event by an audio cue, a visually-rendered text equivalent in the status bar could satisfy this checkpoint. Per checkpoint 6.5, a text equivalent for each such message must be available through an API . See also checkpoint 6.6 for requirements for programmatic notification of changes to the user interface.

Guideline 2. Ensure user access to all content.

Ensure that users have access to all content, notably conditional content that may have been provided to meet the requirements of the Web Content Accessibility Guidelines 1.0 [WCAG10].

The checkpoints in this section require the user agent to provide access to all content through a series of complementary mechanisms designed so that if one fails, another will provide some access. The following preferences are embodied in the checkpoints:

- Both manual and automatic selection of which conditional content to render are important to accessibility.
- Both structured navigation and unstructured access to content are important to accessibility.
- Rendering according to format specification is preferred, but a source view of text content may be necessary for access (e.g., because of user-side error conditions, authoring errors, inadequate specification, or incorrect user agent implementation). For example, in order to find necessary information, the user may have to look at Uniform Resource Identifiers (URIs) for information, HTML comments, XML element names, or script data.

- Configuration and control of rendering are important for access. For instance, the user agent should respect authoring synchronization cues for content that changes over time, but also needs to allow the user to control the time intervals when user input might otherwise be possible.

Authors may use the conditional content mechanisms of a specification to satisfy the requirements of the Web Content Accessibility Guidelines 1.0 *[WCAG10]* . Ensuring access to conditional content benefits all users since some users may not have access to some content due to a technological limitation (e.g., their mobile browser cannot display graphics) or simply a configuration preference (e.g., they have a slow Internet connection and prefer not to download movies or images).

Checkpoint definitions

2.1 Render content according to specification. (P1)

1. Render content according to format specification (e.g., for a markup language or style sheet language).

Normative inclusions and exclusions

1. Rendering requirements include format-defined interactions between author preferences and user preferences/capabilities (e.g., when to render the "`alt`" attribute in HTML, the rendering order of nested `OBJECT` elements in HTML, test attributes in SMIL, and the cascade in CSS2).
2. When a rendering requirement of another specification contradicts a requirement of UAAG 1.0, the user agent may disregard the rendering requirement of the other specification and still satisfy this checkpoint; see the section on the relation of this document to general software design guidelines and other specifications . for more information.
3. The user agent is not required to satisfy this checkpoint for all implemented specifications; see the section on conformance profiles for more information.
4. This checkpoint excludes the requirements of checkpoint 2.6.

Note: If a conforming user agent does not render a content type, it should allow the user to choose a way to handle that content (e.g., by launching another application, by saving it to disk, etc.).

2.2 Provide text view. (P1)

1. For content authored in text formats, provide a view of the text source . For the purposes of this checkpoint, a text format is any media object given an Internet media type of "text" (e.g., "text/plain", "text/html", or "text/*") as defined in RFC 2046 *[RFC2046]* , section 4.1.

Sufficient techniques

1. A user agent satisfies this checkpoint by providing a source view for any text format, not just implemented text formats.

Normative inclusions and exclusions

1. The user agent is only required to satisfy this checkpoint for text formats that are part of a conformance claim; see the section on conformance profiles for more information. However, user agents should provide a text view for all implemented text formats.

2.3 Render conditional content. (P1)

1. Allow configuration to provide access to each piece of unrendered conditional content "C".
2. When a specification does not explain how to provide access to this content, do so as follows:
   - If C is a summary, title, alternative, description, or expansion of another piece of content D, provide access through at least one of the following mechanisms:
     - (1a) render C in place of D;
     - (2a) render C in addition to D;
     - (3a) provide access to C by allowing the user to query D. In this case, the user agent must also alert the user, on a per-element basis, to the existence of C (so that the user knows to query D);
     - (4a) allow the user to follow a link to C from the context of D.
   - Otherwise, provide access to C through at least one of the following mechanisms:
     - (1b) render a placeholder for C, and allow the user to view the original author-supplied content associated with each placeholder;
     - (2b) provide access to C by query (e.g., allow the user to query an element for its attributes). In this case, the user agent must also alert the user, on a per-element basis, to the existence of C;
     - (3b) allow the user to follow a link in context to C.

Sufficient techniques

1. To satisfy provision one of this checkpoint, the configuration may be a switch that, for all content, turns on or off the access mechanisms described in provision two.
2. To satisfy provision two of this checkpoint, the user agent may provide access on a per-element basis (e.g., by allowing the user to query individual elements) or for all elements (e.g., by offering a configuration to render conditional content all the time).

Normative inclusions and exclusions

1. Conformance detail: For all content.

Note: For instance, an HTML user agent might allow users to query each element for access to conditional content supplied for the "`alt`", "`title`", and "`longdesc`" attributes. Or, the user agent might allow configuration so that the value of the "`alt`" attribute is rendered in place of all `IMG` elements (while other conditional content might be made available through another mechanism).

2.4 Allow time-independent interaction. (P1)

1. For rendered content where user input is only possible within a finite time interval controlled by the user agent, allow configuration to provide a view where user interaction is time-independent.

Sufficient techniques

1. The user agent may satisfy this checkpoint by pausing processing automatically to allow for user input, and resuming processing on explicit user request . When this technique is used, pause at the end of each time interval where user input is possible. In the paused state:
   - Alert the user that the rendered content has been paused (e.g., highlight the pause button in a multimedia player's control panel).
   - Highlight which enabled elements are time-sensitive.
   - Allow the user to interact with the enabled elements .
   - Allow the user to resume on explicit user request (e.g., by pressing the play button in a multimedia player's control panel; see also checkpoint 4.5).
2. The user agent may satisfy this checkpoint by generating a time-independent (or, "static") view, based on the original content , that offers the user the same opportunities for interaction. The static view should reflect the structure and flow of the original time-sensitive presentation; orientation cues will help users understand the context for various interaction opportunities.

Normative inclusions and exclusions

1. When satisfying this checkpoint for a real-time presentation, the user agent may discard packets that continue to arrive after the construction of the time-independent view (e.g., when paused or after the construction of a static view).
2. This checkpoint does not apply when the user agent cannot recognize the time interval in the presentation format, or when the user agent cannot control the timing (e.g., because it is controlled by the server).

Note: If the user agent satisfies this checkpoint by pausing automatically, it may be necessary to pause more than once when there are multiple opportunities for time-sensitive user interaction. When pausing, pause synchronized content as well (whether rendered in the same or different viewports) per checkpoint 2.6. In SMIL

1.0 *[SMIL]* , for example, the "`begin`", "`end`", and "`dur`" attributes synchronize presentation components. See also checkpoint 3.5, which involves client-driven content retrieval.

2.5 Make captions, transcripts, audio descriptions available. (P1)

1. Allow configuration or control to render text transcripts , collated text transcripts , captions , and audio descriptions in content at the same time as the associated audio tracks and visual tracks .

Normative inclusions and exclusions

1. Conformance profile labels : Video , Audio .
2. Conformance detail: For all content.

2.6 Respect synchronization cues. (P1)

1. Respect synchronization cues (e.g., in markup) during rendering.

Normative inclusions and exclusions

1. This checkpoint is mutually exclusive of checkpoint 2.1 since it may be excluded from a conformance profile .
2. Conformance profile labels : Video , Audio .

2.7 Repair missing content. (P2)

1. Allow configuration to generate repair text when the user agent recognizes that the author has failed to provide conditional content that was required by the format specification.

Sufficient techniques

1. The user agent may satisfy this checkpoint by basing the repair text on any of the following available sources of information: URI reference, content type, or element type. Note, however, that additional information that would enable more helpful repair might be available but not "near" the missing conditional content. For instance, instead of generating repair text on a simple URI reference, the user agent might look for helpful information near a different instance of the URI reference in the same document object, or might retrieve useful information (e.g., a title) from the resource designated by the URI reference.

Normative inclusions and exclusions

1. Conformance detail: For all content.

Note: Some markup languages (such as HTML 4 *[HTML4]* and SMIL 1.0 *[SMIL]* require the author to provide conditional content for some elements (e.g., the "`alt`" attribute on the `IMG` element).

2.8 No repair text. (P3)

1. Allow at least two configurations for when the user agent recognizes that conditional content required by the format specification is present but empty content :
   - generate no repair text , *or*
   - generate repair as described in checkpoint 2.7.

Normative inclusions and exclusions

1. Conformance detail: For all content.

Note: In some authoring scenarios, empty content (e.g., `alt=""` in HTML) may make an appropriate text equivalent , such as when non-text content has no other function than pure decoration, or when an image is part of a "mosaic" of several images and does not make sense out of the mosaic. Refer to the Web Content Accessibility Guidelines 1.0 *[WCAG10]* for more information about text equivalents.

2.9 Render conditional content automatically. (P3)

1. Allow configuration to render all conditional content automatically.
2. As part of satisfying provision one of this checkpoint, provide access according to specification, or where unspecified, by applying one of the techniques 1a, 2a, or 1b defined in checkpoint 2.3.

Normative inclusions and exclusions

1. The user agent is not required to render all conditional content at the same time in a single viewport.
2. Conformance detail: For all content.

Note: For instance, an HTML user agent might allow configuration so that the value of the "`alt`" attribute is rendered in place of all `IMG` elements (while other conditional content might be made available through another mechanism). The user agent may offer multiple configurations (e.g., a first configuration to render one type of conditional content automatically, a second to render another type, etc.).

2.10 Don’t render unsupported language. (P3)

1. Allow configuration not to render text in unsupported scripts (i.e., writing systems) when that text would otherwise be rendered.
2. When configured per provision one of this checkpoint, indicate to the user in context that author-supplied content has not been rendered.

Normative inclusions and exclusions

1. This checkpoint does not require the user agent to allow different configurations for different natural languages.

Note: This checkpoint is designed primarily to benefit users with serial access to content or who navigate sequentially , allowing them to skip portions of content that would be unusable if rendered as "garbage".

Guideline 3. Allow configuration not to render some content that may reduce accessibility.

Ensure that the user may turn off rendering of content (audio, video, scripts, etc.) that may reduce accessibility by obscuring other content or disorienting the user.

Some content or behavior specified by the author may make the user agent unusable or may obscure information. For instance, flashing content may trigger seizures in people with photosensitive epilepsy, or may make a Web page too distracting to be usable by someone with a cognitive disability. Blinking text can affect screen reader users, since screen readers (in conjunction with speech synthesizers or braille displays) may re-render the text every time it blinks. Distracting background images, colors, or sounds may make it impossible for users to see or hear other content. Dynamically changing Web content may cause problems for some assistive technologies . Scripts that cause unanticipated changes (viewports that open, automatic content retrieval, etc.) may disorient some users with cognitive disabilities.

This guideline requires the user agent to allow configuration so that, when loading Web resources , the user agent does not render content in a manner that might pose accessibility problems. Requirements for interactive control of rendered content are part of guideline 4.

Checkpoint definitions

3.1 Toggle background images. (P1)

1. Allow configuration not to render background image content .

Sufficient techniques

1. The user agent may satisfy this checkpoint with a configuration to not render *any* images, including background images. However, user agents should satisfy this checkpoint by allowing users to turn off background images alone, independent of other types of images in content .

Normative inclusions and exclusions

1. This checkpoint must be satisfied for all implemented image specifications; see the section on conformance profiles .
2. When configured not to render background images, the user agent is not required to retrieve them until the user requests them explicitly. When background images are not rendered, user agents should render a solid background color instead; see checkpoint 4.3 for information about text colors.
3. This checkpoint only requires control of background images for "two-layered renderings", i.e., one rendered background image with all other content rendered "above it".
4. Conformance profile labels : Image .

Note: When background images are not rendered, they are considered conditional content . See checkpoint 2.3 for information about providing access to conditional content.

3.2 Toggle audio, video, animated images. (P1)

1. Allow configuration not to render audio, video, or animated image content , except on explicit user request .

Sufficient techniques

1. The user agent may satisfy this checkpoint by making video and animated images invisible and audio silent , but this technique is not recommended.

Normative inclusions and exclusions

1. This configuration is required for content rendered without any user interaction (including content rendered on load or as the result of a script), as well as content rendered as the result of user interaction that is not an explicit user request (e.g., when the user activates a link).
2. This checkpoint must be satisfied for all implemented audio, video, and animated image specifications; see the section on conformance profiles .
3. When configured not to render audio, video, or animated images except on explicit user request, the user agent is not required to retrieve them until the user requests them explicitly.
4. Conformance profile labels : Animation , Video , Audio .

Note: See guideline 4 for additional requirements related to the control of rendered audio, video, and animated images. When these content types are not rendered, they are considered conditional content . See checkpoint 2.3 for information about providing access to conditional content.

3.3 Toggle animated or blinking text. (P1)

1. Allow configuration to render animated or blinking text content as motionless, unblinking text. Blinking text is text whose visual rendering alternates between visible and invisible, at any rate of change.

Sufficient techniques

1. In this configuration, the user must still have access to the same text content, but the user agent may render it in a separate viewport (e.g., for large amounts of streaming text).
2. The user agent may satisfy this checkpoint by always rendering animated or blinking text as motionless, unblinking text.

Normative inclusions and exclusions

1. This checkpoint must be satisfied for all implemented specifications that support blinking; see the section on conformance profiles .
2. This checkpoint does not apply for blinking and animation effects that are caused by mechanisms that the user agent cannot recognize .
3. Conformance profile labels : VisualText .

Note: Animation (a rendering effect) differs from streaming (a delivery mechanism). Streaming content might be rendered as an animation (e.g., an animated stock ticker or vertically scrolling text) or as static text (e.g., movie subtitles, which are rendered for a limited time, but do not give the impression of movement).

3.4 Toggle scripts. (P1)

1. Allow configuration not to execute any executable content (e.g., scripts and applets).

Normative inclusions and exclusions

1. This checkpoint does not apply to plug-ins and other programs that are not part of content .

Note: Scripts and applets may provide very useful functionality, not all of which causes accessibility problems. Developers should not consider that the user's ability to turn off scripts is an effective way to improve content accessibility; turning off scripts means losing the benefits they offer. Instead, developers should provide users with finer control over user agent or content behavior known to raise accessibility barriers. The user should only have to turn off scripts as a last resort.

3.5 Toggle automatic content retrieval. (P1)

1. Allow configuration so that the user agent only retrieves content on explicit user request .

Normative inclusions and exclusions

1. When the user chooses not to retrieve (fresh) content, the user agent may ignore that content; buffering is not required.
2. The user agent is not required to satisfy this checkpoint for "client-side redirects", i.e., author-specified instructions that a piece of content is temporary and intermediate, and is replaced by content that results from a second request. Authors (and Webmasters) should use the redirect mechanisms of HTTP instead of client-side redirects.
3. This checkpoint only applies when the user agent (not the server) automatically initiates the request for fresh content.

Note: For example, if an HTML author has used a `META` element for automatic content retrieval, allow configuration to override the automatic behavior with manual confirmation.

3.6 Toggle images. (P2)

1. Allow configuration not to render image content .

Sufficient techniques

1. The user agent may satisfy this checkpoint by making images invisible , but this technique is not recommended.

Normative inclusions and exclusions

1. This checkpoint must be satisfied for all implemented image specifications; see the section on conformance profiles .
2. When configured not to render images, the user agent is not required to retrieve them until the user requests them explicitly.
3. Conformance profile labels : Image .

Note: When images are not rendered, they are considered conditional content . See checkpoint 2.3 for information about providing access to conditional content.

Guideline 4. Ensure user control of rendering.

Ensure that the user can select preferred styles (colors, size of rendered text, synthesized speech characteristics, etc.) from choices offered by the user agent. Allow the user to override author-specified styles and user agent default styles.

Providing access to content (see guideline 2) includes enabling users to configure and control its rendering. Users with low vision may require that text be rendered at a size larger than the size specified by the author or by the user agent's default rendering. Users with color blindness may need to impose or prevent certain color combinations.

For dynamic presentations such as synchronized multimedia presentations created with SMIL 1.0 *[SMIL]* , users with cognitive, hearing, visual, and physical disabilities may not be able to interact with a presentation within the time frame assumed by the author. To make the presentation accessible to these users, user agents rendering multimedia content (audio, video, and other animations), have to allow the user to control the playback rate of this content, and also to stop, start, pause, and navigate it quickly. User agents rendering audio have to allow the user to control the audio volume globally and to allow the user to control distinguishable audio tracks.

User agents with speech synthesis capabilities need to allow users to control various synthesized speech rendering parameters. For instance, some users may not be able to make use of high or low frequencies; these users have to be able to configure their speech synthesizers to use suitable frequencies.

Checkpoint definitions for visually rendered text

4.1 Configure text scale. (P1)

1. Allow global configuration of the scale of visually rendered text content. Preserve distinctions in the size of rendered text as the user increases or decreases the scale.
2. As part of satisfying provision one of this checkpoint, provide a configuration option to override rendered text sizes specified by the author or user agent defaults.
3. As part of satisfying provision one of this checkpoint, offer a range of text sizes to the user that includes at least:
   - the range offered by the conventional utility available in the operating environment that allows users to choose the text size (e.g., the font size), *or*
   - if no such utility is available, the range of text sizes supported by the conventional APIs of the operating environment for drawing text.

Sufficient techniques

1. The user agent may satisfy provision one of this checkpoint through a number of mechanisms, including zoom, magnification, and allowing the user to configure a reference size for rendered text (e.g., render text at 36 points unless otherwise specified). For example, for CSS2 *[CSS2]* user agents, the 'medium' value of the 'font-size' property corresponds to a reference size.

Normative inclusions and exclusions

1. The word "scale" is used in this checkpoint to mean the general size of text.
2. The user agent is not required to satisfy this requirement through proportional scaling. What must hold is that if rendered text A is smaller than rendered text B at one value of this configuration setting, then text A will still be smaller than text B at another value of this configuration setting.
3. Conformance profile labels : VisualText .

4.2 Configure font family. (P1)

1. Allow global configuration of the font family of all visually rendered text content.
2. As part of satisfying provision one of this checkpoint, provide a configuration option to override font families specified by the author or by user agent defaults.
3. As part of satisfying provision one of this checkpoint, offer a range of font families to the user that includes at least:
    - the range offered by the conventional utility available in the operating environment that allows users to choose the font family, *or*
    - if no such utility is available, the range of font families supported by the conventional APIs of the operating environment for drawing text.

Sufficient techniques

1. For text that cannot be rendered properly using the user's preferred font family, the user agent may substitute an alternative font family.

Normative inclusions and exclusions

1. Conformance profile labels : VisualText .

Note: For example, allow the user to specify that all text is to be rendered in a particular sans-serif font family.

4.3 Configure text colors. (P1)

1. Allow global configuration of the foreground and background color of all visually rendered text content.
2. As part of satisfying provision one of this checkpoint, provide a configuration option to override foreground and background colors specified by the author or user agent defaults.

3. As part of satisfying provision one of this checkpoint, offer a range of colors to the user that includes at least:
   - the range offered by the conventional utility available in the operating environment that allows users to choose colors, *or*
   - if no such utility is available, the range of colors supported by the conventional APIs of the operating environment for specifying colors.

Normative inclusions and exclusions

1. Color includes black, white, and greys.
2. Conformance profile labels : VisualText .

Note: User configuration of foreground and background colors may inadvertently lead to the inability to distinguish ordinary text from selected text, focused text, etc. See checkpoint 10.2 for more information about highlight styles.

Checkpoint definitions for multimedia presentations and other presentations that change continuously over time

4.4 Slow multimedia. (P1)

1. Allow the user to slow the presentation rate of rendered audio and animation content (including video and animated images).
2. As part of satisfying provision one of this checkpoint, for a visual track , provide at least one setting between 40% and 60% of the original speed.
3. As part of satisfying provision one of this checkpoint, for a prerecorded audio track including audio-only presentations , provide at least one setting between 75% and 80% of the original speed.
4. When the user agent allows the user to slow the visual track of a synchronized multimedia presentation to between 100% and 80% of its original speed, synchronize the visual and audio tracks (per checkpoint 2.6). Below 80%, the user agent is not required to render the audio track .

Normative inclusions and exclusions

1. The user agent is not required to satisfy this checkpoint for audio and animations whose recognized role is to create a purely stylistic effect. Purely stylistic effects include background sounds, decorative animated images, and effects caused by style sheets.
2. Conformance profile labels : Animation , Audio .

Note: The style exception of this checkpoint is based on the assumption that authors have satisfied the requirements of the "Web Content Accessibility Guidelines 1.0" *[WCAG10]* not to convey information through style alone (e.g., through color alone or style sheets alone).

4.5 Start, stop, pause, and navigate multimedia. (P1)

1. Allow the user to stop, pause, and resume rendered audio and animation content (including video and animated images) that last three or more seconds at their default playback rate.
2. Allow the user to navigate efficiently within audio and animations (including video and animated images) that last three or more seconds at their default playback rate.

Sufficient techniques

1. The user agent may satisfy the navigation requirement of provision two of this checkpoint through forward and backward serial access techniques (e.g., advance five seconds), or direct access techniques (e.g., play starting at the 10-minute mark), or some combination.

Normative inclusions and exclusions

1. When serial access techniques are used to satisfy provision two of this checkpoint, the user agent is not required to play back content during advance or rewind (though doing so may help orient the user).
2. When the user pauses a real-time audio or animation, the user agent may discard packets that continue to arrive during the pause.
3. This checkpoint applies to content that is either rendered automatically (e.g., on load) or on explicit request from the user.
4. The user agent is not required to satisfy this checkpoint for audio and animations whose recognized role is to create a purely stylistic effect; see checkpoint 4.4 for more information about what constitutes a stylistic effect.
5. Conformance profile labels : Animation , Audio .

Note: The lower bound of three seconds is part of this checkpoint since control is not required for brief audio and animation clips, beeps, etc. Respect synchronization cues per checkpoint 2.6.

4.6 Do not obscure captions. (P1)

1. For graphical viewports, allow configuration so that captions synchronized with a visual track in content are not obscured by it.

Sufficient techniques

1. Render captions "on top" of the visual track and, as part of satisfying checkpoint 4.3, allow the user to configure the foreground and background color of the rendered captions text.
2. Render captions and video in separate viewports .

Checkpoint definitions for audio volume control

4.7 Global volume control. (P1)

1. Allow global configuration of the volume of all rendered audio, with an option to override audio volumes specified by the author or user agent defaults.
2. As part of satisfying provision one of this checkpoint, allow the user to choose zero volume (i.e., silent).

Normative inclusions and exclusions

1. This checkpoint must be satisfied for all implemented specifications that produce sound; see the section on conformance profiles .
2. Conformance profile labels : Audio .
3. Conformance detail: For both content and user agent.

Note: User agents should allow configuration of volume through available operating environment mechanisms.

4.8 Independent volume control. (P1)

1. Allow independent control of the volumes of rendered audio content synchronized to play simultaneously.

Normative inclusions and exclusions

1. The user control required by this checkpoint includes the ability to override author-specified volumes for the relevant sources of audio.
2. The user agent is not required to satisfy this checkpoint for audio whose recognized role is to create a purely stylistic effect; see checkpoint 4.4 for more information about what constitutes a stylistic effect.
3. Conformance profile labels : Audio .

Note: The user agent should satisfy this checkpoint by allowing the user to control independently the volumes of all audio sources (e.g., by implementing a general audio mixer type of functionality). See checkpoint 4.10 for information about controlling the volume of synthesized speech.

Checkpoint definitions for synthesized speech rendering

4.9 Configure synthesized speech rate. (P1)

1. Allow configuration of the synthesized speech rate, according to the full range offered by the speech synthesizer.

Normative inclusions and exclusions

1. Conformance profile labels : Speech .

Note: The range of synthesized speech rates offered by the speech synthesizer may depend on natural language.

4.10 Configure synthesized speech volume. (P1)

1. Allow control of the synthesized speech volume, independent of other sources of audio .

Normative inclusions and exclusions

1. The user control required by this checkpoint includes the ability to override author-specified synthesized speech volume.
2. Conformance profile labels : Speech .

Note: See checkpoint 4.8 for information about independent volume control of different sources of audio.

4.11 Configure synthesized speech characteristics. (P1)

1. Allow configuration of synthesized speech characteristics according to the full range of values offered by the speech synthesizer.

Normative inclusions and exclusions

1. Conformance profile labels : Speech .

Note: Some speech synthesizers allow users to choose values for synthesized speech characteristics at a higher abstraction layer, i.e., by choosing from present options that group several characteristics. Some typical options one might encounter include: "voice ("adult male voice", "female child voice", "robot voice", etc.), "pitch", "stress", etc. Ranges for values may vary among speech synthesizers.

4.12 Specific synthesized speech characteristics. (P2)

1. Allow configuration of synthesized speech pitch. Pitch refers to the average frequency of the speaking voice.
2. Allow configuration of synthesized speech pitch range. Pitch range specifies a variation in average frequency.
3. Allow configuration of synthesized speech stress. Stress refers to the height of "local peaks" in the intonation contour of the voice.
4. Allow configuration of synthesized speech richness. Richness refers to the richness or brightness of the voice.

Normative inclusions and exclusions

1. Conformance profile labels : Speech .

Note: This checkpoint is more specific than checkpoint 4.11. It requires support for the voice characteristics listed in the provisions of this checkpoint. Definitions for these characteristics are based on descriptions in section 19 of the Cascading Style Sheets Level 2 Recommendation *[CSS2]* ; refer to that specification for additional informative descriptions. Some speech synthesizers allow users to choose values for synthesized speech characteristics at a higher abstraction layer, i.e., by choosing from present options distinguished by "gender", "age", "accent", etc. Ranges of values may vary among speech synthesizers.

4.13 Configure synthesized speech features. (P2)

1. Provide support for user-defined extensions to the synthesized speech dictionary.
2. Provide support for spell-out: where text is spelled one character at a time, or according to language-dependent pronunciation rules.
3. Allow at least two configurations for speaking numerals: one where numerals are spoken as individual digits, and one where full numbers are spoken.
4. Allow at least two configurations for speaking punctuation: one where punctuation is spoken literally, and one where punctuation is rendered as natural pauses.

Normative inclusions and exclusions

1. Conformance profile labels : Speech .

Note: Definitions for the functionalities listed in the provisions of this checkpoint are based on descriptions in section 19 of the Cascading Style Sheets Level 2 Recommendation *[CSS2]* ; refer to that specification for additional informative descriptions.

Checkpoint definitions related to style sheets

4.14 Choose style sheets. (P1)

1. Allow the user to choose from and apply alternative author style sheets (such as linked style sheets).
2. Allow the user to choose from and apply at least one user style sheet .
3. Allow the user to turn off (i.e., ignore) author and user style sheets.

Normative inclusions and exclusions

1. This checkpoint only applies to user agents that support style sheets.

Note: By definition, the user agent's default style sheet is always present, but may be overridden by author or user styles. Developers should not consider that the user's ability to turn off author and user style sheets is an effective way to improve content accessibility; turning off style sheet support means losing the many benefits they offer. Instead, developers should provide users with finer control over user agent or content behavior known to raise accessibility barriers. The user should only have to turn off author and user style sheets as a last resort.

Guideline 5. Ensure user control of user interface behavior.

Ensure that the user can control the behavior of viewports and user interface controls, including those that may be manipulated by the author (e.g., through scripts).

Control of viewport behavior is important to accessibility. Unexpected changes to the point of regard – what the user is presumed to be viewing – may cause users to lose track of how many viewports are open, which viewport has the current focus , etc. If carried out automatically, these changes might go unnoticed (e.g., by some users with blindness) or be disorienting (e.g., to some users with a cognitive disability). This guideline includes requirements for control of opening and closing viewports, the relative position of graphical viewports, changes to focus, and inadvertent form submissions.

Checkpoint definitions

5.1 No automatic content focus change. (P2)

1. Allow configuration so that if a viewport opens without explicit user request , neither its content focus nor its user interface focus automatically becomes the current focus .

Sufficient techniques

1. To satisfy provision one of this checkpoint, configuration is preferred, but is not required if the content focus can only ever be moved on explicit user request .

5.2 Keep viewport on top. (P2)

1. For graphical user interfaces, allow configuration so that the viewport with the current focus remains "on top" of all other viewports with which it overlaps.

5.3 Manual viewport open only. (P2)

1. Allow configuration so that viewports only open on explicit user request .
2. When configured per provision one of this checkpoint, instead of opening a viewport automatically, alert the user and allow the user to open it with an explicit request (e.g., by confirming a prompt or following a link generated by the user agent).
3. Allow the user to close viewports.

Sufficient techniques

1. To satisfy provision one of this checkpoint, configuration is preferred, but is not required if viewports can only ever open on explicit user request .

Normative inclusions and exclusions

1. If a viewport (e.g., a frame set) contains other viewports, these requirements only apply to the outermost container viewport.
2. User creation of a new viewport (e.g., empty or with a new resource loaded) through the user agent's user interface constitutes an explicit user request.

Note: Generally, viewports open automatically as the result of instructions in content . See also checkpoint 5.1 (for control over changes of focus when a viewport opens) and checkpoint 6.6 (for programmatic notification of changes to the user interface).

5.4 Selection and focus in viewport. (P2)

1. Ensure that when a viewport's selection or content focus changes, it is at least partially in the viewport after the change.

Normative inclusions and exclusions

1. Conformance profile labels : Selection .

Note: For example, if users navigating links move to a portion of the document outside a graphical viewport, the viewport should scroll to include the new location of the focus. Or, for users of audio viewports, allow configuration to render the selection or focus immediately after the change.

5.5 Confirm form submission. (P2)

1. Allow configuration to prompt the user to confirm (or cancel) any form submission.

Sufficient techniques

1. Configuration is preferred, but it not required if forms can only ever be submitted on explicit user request .

Note: Examples of automatic form submission include: script-driven submission when the user changes the state of a particular form control associated with the form (e.g., via the pointing device), submission when all fields of a form have been filled out, and submission when a "mouseover" or "change" event occurs.

Guideline 6. Implement interoperable application programming interfaces.

Implement interoperable interfaces to communicate with other software (e.g., assistive technologies, the operating environment, plug-ins, etc.).

This guideline addresses interoperability between a conforming user agent and other software, in particular assistive technologies . The checkpoints of this guideline require implementation of application programming interfaces (APIs) for communication. There are three types of requirements in this guideline:

1. Requirements for what information must be communicated through an API.
2. Requirements for which APIs or types of APIs must be used to communicate this information.
3. Requirements for additional characteristics of these APIs.

Note: The User Agent Accessibility Guidelines Working Group believes that, in order to promote interoperability between a conforming user agent and *more than one* assistive technology, it is more important to implement conventional APIs than custom APIs, even though custom APIs may offer specialized access.

Checkpoint definitions

6.1 Programmatic access to HTML/XML infoset. (P1)

1. Provide programmatic read access to XML content by making available **all** of the information items defined by the W3C XML Infoset *[INFOSET]* .
2. Provide programmatic read access to HTML content by making available all of the following information items defined by the W3C XML Infoset *[INFOSET]* :
   - Document Information item: children, document element, base URI, charset
   - Element Information items: element-type name, children, attributes, parent
   - Attribute Information items: attribute-type name, normalized value, specified, attribute type, references, owner element
   - Character Information items: character code, parent element
   - Comment Information items: content, parent
3. If the user can modify HTML and XML content ("write access") through the user interface (e.g., through form controls), allow for the same modifications

programmatically.

6.2 DOM access to HTML/XML content. (P1)

1. Provide access to the content required in checkpoint 6.1 by conforming to the following modules of the W3C Document Object Model DOM Level 2 Core Specification *[DOM2CORE]* and exporting bindings for the interfaces they define:
   - for HTML: the Core module.
   - for XML: the Core and XML modules.
2. As part of satisfying provision one of this checkpoint,
   - Export the normative bindings specified in the DOM Level 2 Core Specification *[DOM2CORE]* (namely, for Java *[JAVA]* and ECMAScript *[ECMASCRIPT]* operating environments).
   - For other environments, the bindings exported to satisfy provision one of this checkpoint (e.g., C++ bindings) must be publicly documented.

Normative inclusions and exclusions

1. The user agent is not required to export the bindings outside of the user agent process (though doing so may be useful to assistive technology developers).

Note: Refer to the "Document Object Model (DOM) Level 2 Core Specification" *[DOM2CORE]* for information about HTML and XML versions covered. This checkpoint is stands apart from checkpoint 6.1 to emphasize the distinction between what information is required and how to provide access to that information.

6.3 Programmatic access to non-HTML/XML content. (P1)

1. For content other than HTML and XML, provide structured programmatic read access to content , and write access to those parts of content that the user can modify through the user interface .
2. As part of satisfying provision one of this checkpoint, implement at least one API according to this **API cascade**:
   - The API is defined by a W3C Recommendation, **or** the API is publicly documented and designed to enable interoperability with assistive technologies.
   - If no such API is available, or if available APIs do not enable the user agent to satisfy the requirements,
     - implement at least one publicly documented API to satisfy the requirements, *and*
     - follow operating environment conventions for the use of input and output APIs .

Normative inclusions and exclusions

1. "Structured programmatic access" means access through an API to recognized information items of the content (such as the information items of the XML Infoset *[INFOSET]*). Plain text has little structure, so an API that provides access to it will be correspondingly less complex than an API for XML content. For content more structured than plain text, an API that only provides access to a stream of characters does not satisfy the requirement of providing structured programmatic access. This document does not otherwise define what is sufficiently structured access.
2. An API is considered "available" if the specification of the API is published (e.g., as a W3C Recommendation) in time for integration into a user agent's development cycle.

Note: This checkpoint addresses content not covered by checkpoint 6.1 and checkpoint 6.2.

6.4 Programmatic access to information about rendered content. (P1)

1. For graphical user agents, make available bounding dimensions and coordinates of rendered graphical objects. Coordinates must be relative to the point of origin in the graphical environment (e.g., with respect to the desktop), not the viewport.
2. For graphical user agents, provide access to the following information about each piece of rendered text: font family, font size, and foreground and background colors.
3. As part of satisfying provisions one and two of this checkpoint, implement at least one API according to the API cascade described in provision two of checkpoint 6.3.

Note: User agents should provide programmatic access to additional useful information about rendered content that is not available through the APIs required by checkpoints 6.2 and 6.3, including the correspondence (in both directions) between graphical objects and their source in the document object , and information about the role of each graphical object.

6.5 Programmatic operation of user agent user interface. (P1)

1. Provide programmatic read access to user agent user interface controls , selection , content focus , and user interface focus .
2. Provide programmatic write access for those user agent user interface controls that the user can modify through the user interface.
3. As part of satisfying provisions one and two of this checkpoint, implement at least one API according to the API cascade described in provision two of checkpoint 6.3.

Normative inclusions and exclusions

1. For security reasons, user agents are not required to allow instructions in content to modify user agent user interface controls . See more information on security considerations .
2. Conformance detail: For user agent features.

Note: APIs used to satisfy the requirements of this checkpoint may be independent of a particular operating environment (e.g., the W3C DOM), conventional APIs for a particular operating environment, conventional APIs for programming languages, plug-ins , virtual machine environments, etc. User agent developers are encouraged to implement APIs that allow assistive technologies to interoperate with multiple types of software in a given operating environment (user agents, word processors, spreadsheet programs, etc.), as this reuse will benefit users and assistive technology developers. User agents should always follow operating environment conventions for the use of input and output APIs.

6.6 Programmatic notification of changes. (P1)

1. Provide programmatic notification of changes to content , user agent user interface controls , selection , content focus , and user interface focus .
2. As part of satisfying provision one of this checkpoint, implement at least one API according to the API cascade of provision two of checkpoint 6.3.

Normative inclusions and exclusions

1. The user agent is not required to provide notification of changes in the *rendering* of content (e.g., due to an animation effect or an effect caused by a style sheet) unless the document object to make those changes.
2. Conformance profile labels : Selection .
3. Conformance detail: For both content and user agent.

Note: For instance, provide programmatic notification when user interaction in one frame causes automatic changes to content in another.

6.7 Conventional keyboard APIs. (P1)

1. Implement APIs for the keyboard as follows:
   - Follow operating environment conventions .
   - If no conventions exist, implement publicly documented APIs .

Note: An operating environment may define more than one conventional API for the keyboard. For instance, for Japanese and Chinese, input may be processed in two stages, with an API for each.

6.8 API character encodings. (P1)

1. For an API implemented to satisfy requirements of this document, support the character encodings required for that API.

Normative inclusions and exclusions

1. Conformance detail: For both content and user agent.

Note: Support for character encodings is important so that text is not "broken" when communicated to assistive technologies. For example, the DOM Level 2 Core Specification *[DOM2CORE]* , section 1.1.5 requires that the `DOMString` type be encoded using UTF-16.

6.9 DOM access to CSS style sheets. (P2)

1. For user agents that implement Cascading Style Sheets (CSS), provide programmatic access to style sheets by conforming to the CSS module of the W3C Document Object Model (DOM) Level 2 Style Specification *[DOM2STYLE]* and exporting bindings for the interfaces it defines.
2. As part of satisfying provision one of this checkpoint:
   - Export the normative bindings specified in the CSS module of the DOM) Level 2 Style Specification *[DOM2STYLE]* (namely, for Java *[JAVA]* and ECMAScript *[ECMASCRIPT]* operating environments).
   - For other environments, the bindings exported to satisfy provision one of this checkpoint must be publicly documented.

Normative inclusions and exclusions

1. For the purposes of satisfying this checkpoint, Cascading Style Sheets (CSS) are defined by either CSS Level 1 *[CSS1]* or CSS Level 2 *[CSS2]* .
2. The user agent is not required to export the bindings outside of the user agent process.

Note: Refer to the "Document Object Model (DOM) Level 2 Style Specification" *[DOM2STYLE]* for information about CSS versions covered.

6.10 Timely exchanges through APIs. (P2)

1. For APIs implemented to satisfy the requirements of this document, ensure that programmatic exchanges proceed in a timely manner.

Normative inclusions and exclusions

1. Conformance detail: For both content and user agent.

Note: For example, the programmatic exchange of information required by other checkpoints in this document should be efficient enough to prevent information loss, a risk when changes to content or user interface occur more quickly than the communication of those changes. Timely exchange is also important for the proper synchronization of alternative renderings. The techniques for this checkpoint explain how developers can reduce communication delays. This will help ensure that assistive technologies have timely access to the document object model and other information that is important for providing access.

Guideline 7. Observe operating environment conventions.

Observe operating environment conventions for the user agent user interface, documentation, installation, etc.

Part of user agent accessibility involves following the conventions of the user's operating environment, including:

- following operating environment conventions for user agent user interface design, documentation , and installation.
- incorporating operating environment-level user preferences into the user agent. For instance, some operating systems include settings that allow users to request high-contrast colors (for users with low vision) or graphical rendering of audio cues (for users with hearing disabilities).

Following operating environment conventions also increases predictability for users and for developers of assistive technologies . These guidelines explain what users will expect from the look and feel of the user interface, keyboard conventions, documentation, etc. These guidelines also include information about accessibility features that the user agent should adopt rather than reimplement.

The chapter on conformance explains more on the use of operating environment features as part of conformance .

Checkpoint definitions

7.1 Respect focus and selection conventions. (P1)

1. Follow operating environment conventions that benefit accessibility when implementing the selection , content focus , and user interface focus .

Normative inclusions and exclusions

1. This checkpoint is mutually exclusive of checkpoint 7.3 since it has a higher priority.
2. Conformance profile labels : Selection .

Note: See checkpoints 9.1 and 9.2 for more information about content focus and user interface focus.

7.2 Respect input configuration conventions. (P1)

1. Ensure that default input configurations of the user agent do not interfere with operating environment accessibility conventions (e.g., for keyboard accessibility).

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

Note: Information about operating environment accessibility conventions is available in the Techniques document *[UAAG10-TECHS]* . See checkpoint 11.5 for information about the user agent's default input configuration.

7.3 Respect operating environment conventions. (P2)

1. Follow operating environment conventions that benefit accessibility. In particular, follow conventions that benefit accessibility for user interface design, keyboard configuration, product installation, and documentation .

Normative inclusions and exclusions

1. For the purposes of this checkpoint, an operating environment convention that benefits accessibility is either
   - one identified as such in operating environment design or accessibility guidelines, *or*
   - one that allows the author to satisfy any requirement of the "Web Content Accessibility Guidelines 1.0" *[WCAG10]* or of the current document.
2. This checkpoint excludes the requirements of checkpoints 7.1 and 7.4.
3. Conformance detail: For user agent features.

Note: Some of these conventions (e.g., sticky keys, mouse keys, show sounds, etc.) are discussed in the Techniques document *[UAAG10-TECHS]* .

7.4 Provide input configuration indications. (P2)

1. Follow operating environment conventions to indicate the input configuration .

Normative inclusions and exclusions

1. This checkpoint is mutually exclusive of checkpoint 7.3 to emphasize the importance of consistency in input configurations.
2. Conformance detail: For user agent features.

Note: For example, in some operating environments, when a functionality may be triggered through a menu and through the keyboard, the developer may design the menu entry so that the character of the activating key is also shown. See checkpoint 11.5 for information about the user agent's default input configuration.

Guideline 8. Implement specifications that benefit accessibility.

Support the accessibility features of all implemented specifications. Implement W3C Recommendations when available and appropriate for a task.

Developers should implement open specifications. Conformance to open specifications benefits interoperability and accessibility by making it easier to design assistive technologies (also discussed in guideline 6).

While developers should implement the accessibility features of any specification (checkpoint 8.1), this document recommends conformance to W3C Recommendations in particular (checkpoint 8.2) for several reasons:

- W3C specifications include "built-in" accessibility features.
- W3C specifications undergo early review to ensure that accessibility issues are considered during the design phase. This review includes review from stakeholders in accessibility.
- W3C specifications are developed in a consensus process (refer to the process defined by the W3C Process Document *[W3CPROCESS]*). W3C encourages the public to review and comment on these specifications (public Working Drafts, Candidate Recommendations, and Proposed Recommendations). For information about how specifications become W3C Recommendations, refer to the W3C Recommendation track (*[W3CPROCESS]* , section 6.2). W3C Recommendations (and other technical reports) are published at the W3C Web site.

Checkpoint definitions

8.1 Implement accessibility features. (P1)

1. Implement the accessibility features of specifications (markup languages, style sheet languages, metadata languages, graphics formats, etc.).

Normative inclusions and exclusions

1. This checkpoint applies to both W3C-developed and non-W3C specifications.
2. For the purposes of this checkpoint, an accessibility feature of a specification is either:
   - one identified as such in the specification, *or*
   - one that allows the author to satisfy any requirement of the "Web Content

Accessibility Guidelines 1.0" *[WCAG10]* .

3. The user agent is not required to satisfy this checkpoint for all implemented specifications; see the section on conformance profiles for more information.
4. Conformance detail: For all content.

Note: The Techniques document *[UAAG10-TECHS]* provides information about the accessibility features of some specifications, including W3C specifications.

8.2 Conform to specifications. (P2)

1. Use and conform to either
   - W3C Recommendations when they are available and appropriate for a task, *or*
   - non-W3C specifications that enable the creation of content that conforms at level A or better to the Web Content Accessibility Guidelines 1.0 *[WCAG10]* .

Sufficient techniques

1. When a requirement of another specification contradicts a requirement of the current document, the user agent may disregard the requirement of the other specification and still satisfy this checkpoint.

Normative inclusions and exclusions

1. A specification is considered available if it is published (e.g., as a W3C Recommendation) in time for integration into a user agent's development cycle.
2. The user agent is not required to satisfy this checkpoint for all implemented specifications; see the section on conformance profiles for more information.
3. Conformance detail: For all content.

Note: For instance, for markup, the user agent may conform to HTML 4 *[HTML4]* , XHTML 1.0 *[XHTML10]* , and/or XML 1.0 *[XML]* . For style sheets, the user agent may conform to CSS (*[CSS1]* , *[CSS2]*). For mathematics, the user agent may conform to MathML 2.0 *[MATHML20]* . For synchronized multimedia, the user agent may conform to SMIL 1.0 *[SMIL]* .

Guideline 9. Provide navigation mechanisms.

Provide access to content through a variety of navigation mechanisms: sequential navigation, direct navigation, searches, structured navigation, etc.

Users should be able to navigate to important pieces of content within a configurable view, identify the type of object they have navigated to, interact with that object easily (if it is an enabled element), and review the surrounding context (to orient themselves). Providing a variety of navigation and search mechanisms helps users with disabilities (and all users) access content more efficiently. Navigation and searching are particularly important to users with serial access to content or who navigate sequentially .

Direct navigation (e.g., to a particular link or paragraph) is faster than sequential navigation , but generally requires familiarity with the content. Direct navigation is important to users with some physical disabilities (who may have little or no manual dexterity and/or increased tendency to push unwanted buttons or keys), to users with visual disabilities, and also benefits "power users." Direct navigation may be possible with the pointing device or the keyboard (e.g., keyboard shortcuts).

Structured navigation mechanisms offer both context and speed. User agents should allow users to navigate to content known to be structurally important: blocks of content, headers and sections, tables, forms and form elements, enabled elements, navigation mechanisms, containers, etc. For information about programmatic access to document structure, see guideline 6.

User agents should allow users to configure navigation mechanisms (e.g., to allow navigation of links only, or links and headings, or tables and forms, etc.).

Checkpoint definitions

9.1 Provide content focus. (P1)

1. Provide at least one content focus for each viewport (including frames) where enabled elements are part of the rendered content .
2. Allow the user to make the content focus of each viewport the current focus .

Normative inclusions and exclusions

1. When a viewport includes no enabled elements (either because the format does not provide for this, or a given piece of content has no enabled elements), the content focus requirements of the following checkpoints do not apply: 1.2, 5.1, 5.4, 6.6, 7.1, 9.3, 9.4, 9.5, 9.6, 9.7, 10.2, and 11.5.

Note: For example, when two frames of a frameset contain enabled elements, allow the user to make the content focus of either frame the current focus. Note that viewports "owned" by plug-ins that are part of a conformance claim are also covered by this checkpoint. See checkpoint 7.1 for information about implementing content focus according to operating environment conventions.

9.2 Provide user interface focus. (P1)

1. Provide a user interface focus .

Note: See checkpoint 7.1 for information about implementing user interface focus according to operating environment conventions.

9.3 Move content focus. (P1)

1. Allow the user to move the content focus to any enabled element in the viewport .
2. Allow configuration so that the content focus of a viewport only changes on explicit user request .
3. If the author has not specified a navigation order, allow at least forward sequential navigation , in document order, to each element in the set established by provision one of this checkpoint.

Sufficient techniques

1. To satisfy provision one of this checkpoint, configuration is preferred, but is not required if the content focus only ever changes on explicit user request .

Normative inclusions and exclusions

1. The user agent may also include disabled elements in the navigation order.

Note: In addition to forward sequential navigation, the user agent should also allow reverse sequential navigation . See checkpoint 9.9 for information about structured navigation. See checkpoints 5.1 and 6.6 for more information about focus changes.

9.4 Restore viewport state history. (P1)

1. For user agents that implement a viewport history mechanism, for each state in a viewport's browsing history, maintain information about the point of regard , content focus , and selection .
2. When the user returns to any state in the viewport history (e.g., via the "back button"), restore the saved values for the point of regard , content focus , and selection .

Normative inclusions and exclusions

1. The viewport history associates values for these three state variables (point of regard , content focus , and selection) with a particular document object. If the user returns to a state in the history and the user agent retrieves new content, the user agent is not required to restore the saved values of the three state variables.
2. Conformance profile labels : Selection .

9.5 No events on focus change. (P2)

1. Allow configuration so that moving the content focus to or from an enabled element does not automatically activate any explicitly associated event handlers of any event type.

Normative inclusions and exclusions

1. Conformance profile labels : Events .

Note: For instance, in this configuration for an HTML document, do not activate any handlers for the 'onfocus', 'onblur', or 'onchange' attributes. In this configuration, user agents should still apply any stylistic changes (e.g., highlighting) that may occur when there is a change in content focus .

9.6 Show event handlers. (P2)

1. For the element with content focus , make available the list of input device event types for which there are event handlers explicitly associated with the element.

Normative inclusions and exclusions

1. Conformance profile labels : Events .

Note: For example, allow the user to query the element with content focus for the list of input device event types, or add them directly to the sequential navigation order described in checkpoint 9.3. See checkpoint 1.2 for information about activation of event handlers associated with the element with focus.

9.7 Move content focus in reverse. (P2)

1. Extend the functionality required in provision three of checkpoint 9.3 by allowing the same sequential navigation in reverse document order.
2. As part of satisfying provision one of this checkpoint, the user agent must not include disabled elements in the navigation order.

9.8 Provide text search. (P2)

1. Allow the user to search within rendered text content for a sequence of characters from the document character set .
2. Allow the user to start a forward search (in document order) from any selected or focused location in content.
3. When there is a match, do both of the following:
   - move the viewport so that the matched text content is within it, *and*
   - allow the user to search for the next instance of the text from the location of the match.
4. Alert the user when there is no match or after the last match in content (i.e., prior to starting the search over from the beginning of content).
5. Provide a case-insensitive search option for text in scripts (i.e., writing systems) where case is significant.

Normative inclusions and exclusions

1. Conformance detail: For all rendered content.

Note: If the user has not indicated a start position for the search, the search should start from the beginning of content. Per checkpoint 7.3, use operating environment conventions for indicating the result of a search (e.g., selection or content focus).

9.9 Allow structured navigation. (P2)

1. Allow the user to navigate efficiently to and among important structural elements in rendered content .
2. As part of satisfying provision one of this checkpoint, allow forward and backward sequential navigation .

Note: This specification intentionally does not identify which "important elements" must be navigable as this will vary by specification. What constitutes "efficient navigation" may depend on a number of factors as well, including the "shape" of content (e.g., sequential navigation of long lists is not efficient) and desired granularity (e.g., among tables, then among the cells of a given table). Refer to the Techniques document *[UAAG10-TECHS]* for information about identifying and navigating important elements.

9.10 Configure important elements. (P3)

1. Allow configuration of the set of important elements and attributes identified for checkpoints 9.9 and 10.4.
2. As part of satisfying provision one of this checkpoint, allow the user to include and exclude element types in the set.

Note: For example, allow the user to navigate only paragraphs, or only headings and paragraphs, or to suppress and restore navigation bars, to navigate within and among tables and table cells, etc.

Guideline 10. Orient the user.

Provide information that will help the user understand browsing context.

All users require clues to help them understand their "location" when browsing: where they are, how they got there, where they can go, what’s nearby, etc. Some mechanisms that provide such clues through the user interface (visually, as audio, or as braille) include:

- information about the current state of the user’s interaction with content: where the viewport is in content (shown, for example, through proportional scroll bars), which viewport has the current focus , where the user has selected content, a history mechanism, the title of the current document or frame, etc.

- information about specific elements, such as the dimensions of a table, the length of an audio clip, the structure of a form, etc.
- information about relationships among elements, such as between table cells and related table headers.
- information about the structure of content, e.g., through an outline view of a document.

Orientation mechanisms such as these are especially important to users with serial access to content or who navigate sequentially . For instance, these users cannot "scan" a graphically displayed table with their eyes for information about a table cell's headers, neighboring cells, etc. User agents need to provide other means for users to understand table cell relationships, frame relationships (what relationship does the graphical layout convey?), form context (have I filled out the form completely?), link information (have I already visited this link?), etc.

Checkpoint definitions

10.1 Associate table cells and headers. (P1)

1. For graphical user agents that render tables, for each table cell, allow the user to view associated header information.

Sufficient techniques

1. The user agent may satisfy this checkpoint by allowing the user to query each table cell for associated header information.
2. The user agent may satisfy this checkpoint by rendering the table cell and associated header information so they are both visible in the same viewport.

Normative inclusions and exclusions

1. This checkpoint refers only to cell/header relationships that the user agent can recognize .

10.2 Highlight selection, content focus, enabled elements, visited links. (P1)

1. Allow global configuration to highlight the following four classes of information in each viewport: the selection , content focus , enabled elements , and recently visited links.
2. For graphical user interfaces, as part of satisfying provision one of this checkpoint, allow at least one configuration where the highlight mechanisms for the four classes of information:
   - differ from each other, *and*
   - do not rely on rendered text foreground and background colors alone.
3. For graphical user interfaces, as part of satisfying provision one of this checkpoint, if a highlight mechanism involves text size, font family, rendered text

foreground and background colors, or text decorations, offer at least the following range of values:
- for text size, the range required by provision three of checkpoint 4.1.
- for font family, the range required by provision three of checkpoint 4.2.
- for text foreground and background colors and decorations, the range offered by the conventional utility available in the operating environment for users to choose rendered text colors or decorations (e.g., the standard font and color dialog box resources supported by the operating system). If no such utility is available, the range supported by the conventional APIs of the operating environment for specifying text colors or drawing text.

4. Highlight enabled elements according to the granularity specified in the format. For example, an HTML user agent rendering a PNG image as part of a client-side image map is only required to highlight the image as a whole, not each enabled region. An SVG user agent rendering an SVG image with embedded graphical links is required to highlight each (enabled) link that may be rendered independently according to the SVG specification.

Normative inclusions and exclusions

1. Conformance profile labels : Selection .

Note: Examples of highlight mechanisms for selection and content focus include foreground and background color variations, underlining, distinctive synthesized speech prosody, border styling, etc. Because the selection and focus change frequently, user agents should not highlight them using mechanisms (e.g., font size variations) that cause content to reflow, as this may disorient the user. Graphical highlight mechanisms that generally do not rely on rendered text foreground and background color alone include underlines or border styling. Per checkpoint 7.1, follow operating environment conventions that benefit accessibility when implementing the selection and content focus. For instance, if specified at the level of the operating environment, inherit the user's preferences for selection styles.

10.3 Single highlight configuration. (P2)

1. Extend the functionality required by provision two of checkpoint 10.2 by allowing configuration through a single setting.

Normative inclusions and exclusions

1. Conformance profile labels : Selection .

10.4 Provide outline view. (P2)

1. Make available to the user an "outline" view of rendered content , composed of labels for important structural elements (e.g., heading text, table titles, form titles, and other labels that are part of the content).

Normative inclusions and exclusions

1. What constitutes a label is defined by each markup language specification. For example, in HTML, a heading (`H1`-`H6`) is a label for the section that follows it, a `CAPTION` is a label for a table, the "`title`" attribute is a label for its element, etc.
2. The user agent is not required to generate a label for an important element when no label is present in content. The user agent may generate a label when one is not present.
3. A label is not required to be text only.

Note: This outline view will provide the user with a simplified view of content (e.g, a table of contents). For information about what constitutes the set of important structural elements, see the Note following checkpoint 9.9. By making the outline view navigable, it is possible to satisfy this checkpoint and checkpoint 9.9 together: allow users to navigate among the important elements of the outline view, and to navigate from a position in the outline view to the corresponding position in a full view of content. See checkpoint 9.10 for additional configuration options.

10.5 Provide link information. (P3)

1. To help the user decide whether to traverse a link in content , make available the following information about it:
   - link element content,
   - link title,
   - whether the link is internal to the resource (e.g., the link is to a target in the same Web page),
   - whether the user has traversed the link recently, *and*
   - information about the type, size, and natural language of linked Web resources.

Normative inclusions and exclusions

1. The user agent is not required to compute or make available information that requires retrieval of linked Web resources .

Checkpoint definitions for the user interface

10.6 Highlight current viewport. (P1)

1. Highlight the viewport with the current focus (including any frame that takes current focus).
2. For graphical viewports, as part of satisfying provision one of this checkpoint, provide at least one highlight mechanism that does not rely on rendered text foreground and background colors alone (e.g., use a thick outline).
3. If the techniques used to satisfy provision one of this checkpoint involve rendered text size, font family, rendered text foreground and background

colors, or text decorations, allow global configuration and offer same ranges of values required by provision three of checkpoint 10.2.

Note: See checkpoint 7.1 for information about implementing highlight mechanisms according to operating environment conventions.

10.7 Indicate viewport position. (P3)

1. Indicate the viewport's position relative to rendered content (e.g., the proportion of an audio or video clip that has been played, the proportion of a Web page that has been viewed, etc.).

Sufficient techniques

1. The user agent may calculate the relative position according to content focus position, selection position, or viewport position, depending on how the user has been browsing.
2. The user agent may indicate the proportion of content viewed in a number of ways, including as a percentage, as a relative size in bytes, etc. See checkpoint 1.3 for more information about text versions of messages to the user, including messages about position information.

Normative inclusions and exclusions

1. For two-dimensional spatial renderings, relative position includes both vertical and horizontal positions.
2. This checkpoint does not require the user agent to present information about *retrieval* progress. However, for streaming content, viewport position may be closely tied to retrieval progress.

Guideline 11. Allow configuration and customization.

Allow users to configure the user agent so that frequently performed tasks are made convenient, and allow users to save their preferences.

Web users have a wide range of capabilities and need to be able to configure the user agent according to their preferences for styles, graphical user interface configuration, keyboard configuration, etc. Most of the checkpoints in this guideline pertain to the input configuration: how user agent behavior is controlled through keyboard input, pointing device input, and voice input. An input configuration is the set of "bindings" between user agent functionalities and user interface input mechanisms.

The chapter on conformance explains more about configuration requirements and conformance .

Checkpoint definitions

11.1 Current user input configuration. (P1)

1. Provide information to the user about current user preferences for input configurations .

Sufficient techniques

1. To satisfy this checkpoint, the user agent may make available binding information in a centralized fashion (e.g., a list of bindings) or a distributed fashion (e.g., by listing keyboard shortcuts in user interface menus). See related documentation checkpoints 12.2, 12.3, and 12.5.

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

11.2 Current author input configuration. (P2)

1. Provide a centralized view of the current author-specified input configuration .

Sufficient techniques

1. The user agent may satisfy this checkpoint by providing different views for different input modalities (keyboard, pointing device, voice, etc.).

Normative inclusions and exclusions

1. Conformance detail: For all content.

Note: For example, for HTML documents, provide a view of keyboard bindings specified by the author through the "`accesskey`" attribute. The intent of this checkpoint is to centralize information about author-specified bindings so that the user does not have to read an entire document to look for available bindings.

11.3 Allow override of bindings. (P2)

1. Allow the user to override any binding that is part of the user agent default input configuration .

Normative inclusions and exclusions

1. The user agent is not required to allow the user to override conventional bindings for the operating environment (e.g., for access to help).
2. The override requirement only applies to bindings for the same input modality (e.g., the user must be able to override a keyboard binding with another keyboard binding).
3. This checkpoint excludes the requirements of checkpoint 11.4.

4. Conformance detail: For user agent features.

Note: See checkpoint 11.5 for default input configuration requirements and checkpoint 12.3 for information about their documentation.

11.4 Single-key access. (P2)

1. Allow the user to override any binding in the user agent default keyboard configuration with a binding to either a key plus modifier keys or to a single key.
2. For each functionality in the set required by checkpoint 11.5, allow the user to configure a single-key binding. A single-key binding is one where a single key press performs the task, with zero modifier keys.

Sufficient techniques

1. The user agent may satisfy the requirements of provision two of this checkpoint with a "single-key mode" (i.e., a mode where the current bindings are replaced by a set of single-key bindings).

Normative inclusions and exclusions

1. In this checkpoint, "key" refers to a physical key of the keyboard (rather than, say, a character of the document character set).
2. The user agent is not required to allow the user to override conventional bindings for the operating environment (e.g., for access to help).
3. Provision two of this checkpoint does not require single physical key bindings for character input, only for the activation of user agent functionalities.
4. If the number of physical keys on the keyboard is less than the number of functionalities required by checkpoint 11.5, then provision two of this checkpoint does not require the user agent to allow single-key bindings for all of the functionalities. The user agent should give preference to those functionalities listed in provision one of checkpoint 11.5.
5. This checkpoint is mutually exclusive of checkpoint 11.3 since it is specific to the keyboard and to emphasize the importance of easy keyboard access.
6. Conformance detail: For user agent features.

Note: Because single-key access is so important to some users with physical disabilities, user agents should ensure that: (1) most keys of the physical keyboard may be configured for single-key bindings, and (2) most functionalities of the user agent may be configured for single-key bindings. For information about access to user agent functionality through a keyboard API, see checkpoint 6.7.

11.5 Default input configuration. (P2)

1. Ensure that the user agent default input configuration includes bindings for the following functionalities required by other checkpoints in this document:
   - move content focus to the next enabled element in document order, and move content focus to the previous enabled element in document order

(checkpoints 9.3 and 9.7);
   - activate the link designed by the content focus (checkpoints 1.1 and 9.1);
   - search for text, search again for same text (checkpoint 9.8);
   - increase the scale of rendered text , and decrease the scale of rendered text (checkpoint 4.1);
   - increase global volume, and decrease global volume (checkpoint 4.7);
   - stop, pause, resume, and navigate efficiently selected audio and animations , including video and animated images (checkpoint 4.5).
2. If the user agent supports the following functionalities, the default input configuration must also include bindings for them:
   - next history state (forward), and previous history state (back);
   - enter URI for a new resource;
   - add a URI to favorites (i.e., bookmarked resources);
   - view favorites;
   - reload a resource;
   - interrupt a request to reload a resource;
   - for graphical viewports: navigation forward and backward through rendered content by approximately the height of the viewport;
   - for user agents that render content in lines of (at least) text: move point of regard to next line, and previous line.

Sufficient techniques

1. The user agent may satisfy the functionality of entering a URI for a new resource in a number of ways, including by prompting the user or by moving the user interface focus to a control for entering URIs.

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

Note: This checkpoint does not make any requirements about the ease of use of default input configurations, though clearly the default configuration should include single-key bindings and allow easy operation. Ease of use is addressed by the configuration requirements of checkpoint 11.3.

11.6 User profiles. (P2)

1. For the configuration requirements of this document, allow the user to save user preferences in at least one user profile .
2. Allow the user to choose from among available user agent default profiles , profiles created by the same user, and no profile (i.e., the user agent default settings).

Normative inclusions and exclusions

1. This checkpoint does not require the user agent to provide multiple default profiles.
2. This checkpoint does not require that user profiles be portable, i.e., removable from the user agent to be reread by a different instance of the agent. Portable user profiles are very useful, however.
3. Conformance detail: For user agent features.

11.7 Tool bar configuration. (P3)

1. For graphical user agent user interfaces with tool bars, allow the user to configure the position of user agent user interface controls on those tool bars.
2. Offer a predefined set of controls that may be added to or removed from tool bars.
3. Allow the user to restore the default tool bar configuration.

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

Guideline 12. Provide accessible user agent documentation and help.

Ensure that the user can learn about software features that benefit accessibility from the documentation. Ensure that the documentation is accessible.

Documentation of the user interface is important, as is documentation of the user agent’s underlying functionalities. While intuitive user interface design is valuable to many users, some users may still not be able to understand or be able to operate the native user interface without thorough documentation. For instance, a user with blindness may not find a graphical user interface intuitive without supporting documentation.

There are three types of requirements in this guideline:

1. accessibility of the documentation (checkpoint 12.1);
2. minimal requirements of what must be documented (checkpoints 12.2, 12.3, and 12.4). Documentation should include much more to explain how to install, get help for, use, or configure the user agent;
3. organization of the documentation (checkpoint 12.5).

See checkpoint 7.3 for information about following system conventions for documentation.

Checkpoint definitions

12.1 Provide accessible documentation. (P1)

1. Ensure that at least one version of the user agent documentation conforms to at least level Double-A of the Web Content Accessibility Guidelines 1.0 *[WCAG10]* .

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

12.2 Document accessibility features. (P1)

1. Document all user agent features that benefit accessibility.

Sufficient techniques

1. The user agent may satisfy this checkpoint either by
   - providing a centralized view of the accessibility features, or
   - integrating accessibility features into the rest of the documentation.

   A centralized view is sufficient to satisfy this checkpoint and required to satisfy checkpoint 12.5.

Normative inclusions and exclusions

1. For the purposes of this checkpoint, a user agent feature that benefits accessibility is one implemented to satisfy the requirements of this document (including the requirements of checkpoints 8.1 and 7.3, and the API requirements of guideline 6).
2. Conformance detail: For user agent features.

Note: The help system should include discussion of user agent features that benefit accessibility. The user agent should satisfy this checkpoint by providing both centralized and integrated views of accessibility features in the documentation.

12.3 Document default bindings. (P1)

1. Document the default user agent input configuration (e.g., the default keyboard bindings).

Sufficient techniques

1. If the user agent does not allow the user to override the default user agent input configuration (see checkpoint 11.3), the documentation used to satisfy this checkpoint also satisfies checkpoint 11.1.

Normative inclusions and exclusions

1. Conformance detail: For user agent features.

Note: Documentation should warn the user whenever the default input configuration is inconsistent with conventions of the operating environment.

12.4 Document changes between versions. (P2)

1. Document changes from the previous version of the user agent to features that benefit accessibility, including features of the user interface.

Normative inclusions and exclusions

1. The features that benefit accessibility are those defined in checkpoint 12.2.
2. Conformance detail: For user agent features.

12.5 Provide dedicated accessibility section. (P2)

1. Provide a centralized view of all features of the user agent that benefit accessibility, in a dedicated section of the documentation .

Sufficient techniques

1. A centralized view is required to satisfy this checkpoint and sufficient to satisfy checkpoint 12.2.

Normative inclusions and exclusions

1. The features that benefit accessibility are those defined in checkpoint 12.2.
2. Conformance detail: For user agent features.

Note: Developers are encouraged to integrate descriptions of accessibility features into the documentation alongside other features, in addition to providing a centralized view.

3. Conformance

A user agent conforms to this document by satisfying the requirements identified by a conformance profile. This normative section explains:

1. How to construct a conformance profile . A user agent is not required to satisfy every requirement in this document in order to conform.
2. How to make a conformance claim , i.e., a statement about how a chosen user agent satisfies the requirements identified by a chosen conformance profile. See the section on target user agents in the introduction for information about which user agents are expected to conform.
3. How to include UAAG 1.0 requirements in another specification .

Conformance to the requirements of this document is expected to be a strong indicator of accessibility, but it may be neither a necessary nor a sufficient condition for ensuring the accessibility of software. Thus, some software may not conform to this document but still be accessible to some users with disabilities. Conversely, some software may conform to this document but still be inaccessible to some users with disabilities. Some requirements of this document may not benefit some users for some content, but the requirements are expected to benefit many users with disabilities, for general purpose content. For more information, see the section on known limitations of this document , and the section on restricted functionality and conformance .

This document demands substantially more conformance flexibility than can be achieved using the terms "must", "should", and "may" alone, as defined in RFC 2119 *[RFC2119]* . Where "must", "should", "required", and "may" appear in this document, they are used consistently with RFC 2119 for a chosen conformance profile. The imperative voice (e.g., "Allow configuration ...") used in the checkpoint provisions implies "must", but a user agent is only obligated to satisfy the requirements of a chosen conformance profile.

Note: UAAG 1.0 extends significantly the conformance mechanism defined in both WCAG 1.0 *[WCAG10]* and ATAG 1.0 *[ATAG10]* .

3.1 Conformance profiles

A conformance profile is a list of assertions that identify:

1. a version of UAAG 1.0,
2. a set of requirements in that document, and
3. a list of specifications implemented to satisfy some of those requirements.

There are two primary uses for a conformance profile:

- In a conformance claim . When included as part of a conformance claim, a profile generally indicates the "maximum" (most favorable) set of requirements that are satisfied by the user agent.

- In another specification . When included as part of another specification, a profile generally indicates the "minimum" (lower bound) set of requirements that must be satisfied as part of conformance to that specification.

In either case, a conformance profile identifies a set of requirements derived from a default set according to the following mechanisms, collectively called **conformance profile labels**:

1. Conformance levels ,
2. Content type labels ,
3. Events label ,
4. Selection label ,
5. Input modality labels , and
6. Applicability .

The following sections define the default set of requirements, the structure of a conformance profile, and how to determine the set of requirements identified by a profile. UAAG 1.0 does not define any (named) conformance profiles, but rather defines the mechanism for creating them.

3.1.1 Default set of conformance requirements

The **default set** of conformance requirements is defined to be all of the requirements of all of the provisions of all the checkpoints, as qualified by their normative inclusions and exclusions and the following normative inclusions and sufficient techniques that apply *across* checkpoints.

Normative inclusion: User interface or API

Except for the checkpoints in guideline 6 that refer to implementation of APIs , the user agent must satisfy the checkpoint requirements through at least one mechanism other than an API . Thus, for most of the requirements in this document, it is not possible to conform by only making information available through an API (which would be used, for example, by an assistive technology to provide the missing feature). For example, checkpoint 9.3 involves navigation that must be possible through the user interface, not just via an API. This and other checkpoints involving user control or configuration will therefore generally be satisfied through features in the user interface or through configuration files (see the section on configuration requirements).

Normative inclusion: For content, user agent features, or both

In some cases, a checkpoint may apply equally well to content or user agent features. When it is necessary to remove ambiguity about the scope of a checkpoint, the checkpoint definition includes one of the following labels:

1. For content only, i.e., the document object only.
2. For user agent features only, i.e., everything that is not content (such as components of the user agent user interface , user preferences, the user agent

documentation , and the user interface focus).
3. For both content and user agent features.

A user agent may *also* satisfy a "content only" checkpoint for user agent user interface features, and vice-versa. Indeed, user agent developers are encouraged to consider the content-only requirements (e.g., checkpoint 3.3) when designing the user agent's user interface.

The user agent may satisfy a content-only requirement with a mechanism that *also* involves user agent features. For instance, to satisfy checkpoint 4.7, the user agent may provide control for all volume, whether the source is content or the user agent user interface. Similarly, to satisfy checkpoint 3.3, the user agent may offer a single configuration that turns off blinking in both content and the user interface.

Sufficient technique: Configuration files

The user agent may satisfy the configuration requirements of this document through configuration files (e.g., profiles, initialization files, style sheets, themes, etc.). For instance, style sheets might be used as a mechanism to satisfy the highlight and configuration requirements of checkpoint 10.2. Any functionality that is configurable through a configuration file should also be configurable through the user agent user interface . Furthermore, if configuration files may be edited by hand, the user agent documentation should explain the configuration file format, or refer to an explanation (a format specification, for example).

For some of the checkpoints in this document (checkpoints 3.3, 5.1, 5.3, and 5.5), configuration is preferred, but not required to satisfy the checkpoint in some circumstances. For other checkpoints, configurability may be as important as the functionality being configured, and is therefore mandatory.

Since this document allows conformance by a user agent consisting of multiple software components, there are likely to be times when, to satisfy the configuration requirements of the document, each component has to provide for configuration independently. To make configuration easier for the user, components should share and inherit configurations (including those of the operating environment).

Sufficient technique: Multiple operating environments

When a user agent runs in more than one operating environment (e.g., a user agent implemented in Java on top of another operating system), the user agent may satisfy the relevant requirements (e.g., the checkpoints in guideline 7) of a conformance profile by following the conventions of a single operating environment.

When faced with a choice between the conventions of different operating environments, a developer should follow the conventions that benefit accessibility most, while meeting the developer's design goals. For instance, some developers may prefer cross-platform consistency over consistency with other user agents running in a given operating environment, and this might affect which conventions would be preferred.

3.1.2 Parts of a conformance profile

A conformance profile includes the following assertions:

1. Required: The guidelines title/version: "User Agent Accessibility Guidelines 1.0".
2. Required: The URI of the guidelines: http://www.w3.org/TR/2002/WD-UAAG10-20020821.
3. Required: The conformance level satisfied: "A", "Double-A", or "Triple-A".
4. Required: At least one content type label . The VisualText label must be present if the user agent renders text visually.
5. Required: The Selection label , if the user agent implements a selection mechanism.
6. Required: A list of requirements (checkpoints or portions of checkpoints) that do **not** apply . A conformance profile should also explain why those requirements do not apply.
7. Required: Information about one or more specifications (e.g., markup languages, style sheet languages, APIs, etc.) implemented to satisfy the requirements of this document. A user agent must satisfy the requirements identified by the profile for at least these specifications. A user agent is not required to satisfy the identified requirements for other implemented specifications *except* when a content type label definition states otherwise. The profile must include enough information to identify the implemented specifications. The profile should indicate which specifications are used to satisfy which requirements (e.g., which image formats are used to satisfy the requirements associated with the Image content type label).
8. Optional: The Events label .
9. Optional: Input modality labels : "Pointer" and/or "Voice".

A profile should not include other information that the required and optional assertions. The wording of the profile should reflect whether the profile is used as part of a conformance claim ("the user agent satisfies these requirements") or as part of another specification ("the user agent must satisfy these requirements").

When a profile is part of a conformance claim, the absence of a conformance profile label implies that the associated requirements are not be satisfied (though the requirements may or may not actually be satisfied). When a profile is included in another specification, the absence of a conformance profile label implies that the associated requirements need not be satisfied.

Thus, a conformance profile when evaluating a user agent might be as short as:

For "User Agent Accessibility Guidelines 1.0", http://www.w3.org/TR/2002/WD-UAAG10-20020821:

- Conformance level: Double A
- Supported conformance profile labels: VisualText, Image, Animation, Audio, Events, and Selection
- A list of checkpoints that do not apply is available online (link to list)

- The specifications that are part of this profile are W3C's HTML 4.0, CSS2, PNG, and SVG (link to each specification)

An extended example below illustrates how to build a conformance profile while evaluating a user agent.

3.1.3 Requirements identified by a conformance profile

The set of requirements identified by a conformance profile is the set derived from the default set by:

1. Removing the requirements associated with conformance levels that do not appear in the profile, and
2. Removing the requirements associated with content type labels that do not appear in the profile, and
3. Removing the requirements associated with the Events label if it does not appear in the profile, and
4. Removing the requirements associated with the Selection label if it does not appear in the profile, and
5. Adding the requirements associated with the input modality labels if they appear in the profile, and
6. Removing the requirements of any checkpoints or parts of checkpoints that the profile asserts do not apply .

The requirements that cannot be removed through the above mechanisms are part of every UAAG 1.0 conformance profile (including, for example, the keyboard requirements of checkpoint 1.1).

3.1.4 Conformance levels

Each conformance level defines a set of requirements, based on priority .

- **Conformance level "A"**: the requirements of all Priority 1 checkpoints.
- **Conformance level "Double-A"**: the requirements of all Priority 1 and 2 checkpoints.
- **Conformance level "Triple-A"**: the requirements of all Priority 1, 2, and 3 checkpoints.

Note: Conformance levels are spelled out in text (e.g., "Double-A" rather than "AA") so they may be understood when rendered as synchronized speech.

3.1.5 Content type labels

Each content type label defines a set of requirements related to support for visually rendered text, images, animations, video, audio, and synthesized speech.

VisualText
: This content type label refers to all of the requirements related to visually rendered text for the following checkpoints: 3.3, 4.1, 4.2, and 4.3. If a user agent

renders text visually, it must satisfy these requirements in order to conform. An audio-only or tactile-only user agent is not required to satisfy the requirements associated with this label. The user agent must satisfy these requirements for all implemented formats that produce visually rendered text , not just those identified in a conformance profile .

Image
: This content type label refers to all of the requirements related to images (excluding animated images) for the following checkpoints: 3.1 and 3.6. To conform, the user agent must implement at least one image format. The user agent must satisfy these requirements for all implemented image formats, not just those identified in a conformance profile . The image requirements apply to image content that is recognized as distinct and that, according to the encoding format, may be rendered as a coherent unit.

Animation
: This content type label refers to all of the requirements related to animations (including video and animated images) for the following checkpoints: 3.2, 4.4, and 4.5. To conform, the user agent must implement at least one animation format. The user agent must satisfy the requirements of checkpoint 3.2 for all implemented animation formats, not just those identified in a conformance profile . The animation requirements apply to animation content that is recognized as distinct and that, according to the encoding format, may be rendered as a coherent unit.

Video
: This content type label refers to all of the requirements related to video for the following checkpoints: 2.5, 2.6, and 3.2. To conform, the user agent must implement at least one video format. The user agent must satisfy the requirements of checkpoint 3.2 for all implemented video formats, not just those identified in a conformance profile . The video requirements apply to video content that is recognized as distinct and that, according to the encoding format, may be rendered as a coherent unit.

Audio
: This content type label refers to all of the requirements related to audio for the following checkpoints: 2.5, 2.6, 3.2, 4.4, 4.5, 4.7, and 4.8. To conform, the user agent must implement at least one audio format. The user agent must satisfy the requirements of checkpoints 3.2 and 4.7 for all implemented audio formats, not just those identified in a conformance profile . The audio requirements apply to audio content that is recognized as distinct and that, according to the encoding format, may be rendered as a coherent unit.

Speech
: This content type label refers to all of the requirements related to synthesized speech for the following checkpoints: 4.9, 4.10, 4.11, 4.12, and 4.13. To conform, the user agent must support synthesized speech.

Note: As indicated above, some of the content type labels require implementation of at least one format (e.g., for images). This document does not require implementation of specific formats, (e.g., PNG *[PNG]* versus SVG *[SVG]* for images). However, see the requirements of checkpoint 8.2.

Some of the content type labels require that certain checkpoints be satisfied for all implemented specifications, not just those listed in a conformance profile , in order to ensure that the goal of the checkpoint is met. For instance, checkpoint 3.3 involves turning off blinking and animated text. Since there is a risk that these rendering effects may trigger seizures in people with photosensitive epilepsy, it is important that the user be able to turn them off in all cases (whether or not the specification is identified in a conformance profile).

3.1.6 Events label

The following checkpoints are designed to augment user agent support for event-driven behavior specified by the author: 1.2, 9.5, and 9.6. Satisfying these checkpoints will promote input device independence and thus enable users with some disabilities to make better use of content designed for a single input device (generally a pointing device). The Events label refers to the requirements of these checkpoints.

3.1.7 Selection label

This document does not require the user agent to implement a selection mechanism in order to conform. However, if the user agent does implement a selection mechanism, in order to conform it must satisfy the relevant portions of the following checkpoints: 5.4, 6.6, 7.1, 9.4, 10.2, and 10.3. The Selection label refers to the selection requirements of these checkpoints.

Note: This document does require implementation of both content focus and user interface focus ; see checkpoints 9.1 and 9.2.

3.1.8 Input modality labels

Each input modality label defines a set of requirements related to support for a particular type of input device. Input device requirements in this document are either stated generically (e.g., "input configuration" requirements) or as keyboard-specific requirements (e.g., "keyboard API").

Pointer
: This input modality label refers to all of the input device requirements of this document, but applied to pointing device input. For keyboard-specific requirements, substitute "pointing device input" for "keyboard." The set of pointing device input requirements does not include the requirements of checkpoint 11.4.

Voice
: This input modality label refers to all of the input device requirements of this document, but applied to voice input. For keyboard-specific requirements, substitute "voice input" for "keyboard." The set of voice input requirements does not include the requirements of checkpoint 11.4.

Note: Developers are encouraged to design user agents that are at least partially operable through pointing device and/or voice input, in addition to being fully operable through the keyboard.

3.1.9 Checkpoint applicability

A checkpoint (or part of a checkpoint) applies *unless* any one of the following conditions is met:

1. The checkpoint makes requirements for graphical user interfaces or graphical viewports and the user agent only has audio or tactile user interfaces or viewports.
2. The checkpoint refers to a role of content (e.g., transcript, captions, associated conditional content , synchronization cue, a table element, etc.) that the user agent cannot recognize because of how the content has been encoded in a particular format. For instance, HTML user agents can recognize "`alt`", `OBJECT` content, or `NOFRAMES` content as specified mechanisms for conditional content . On the other hand, HTML user agents are not expected to recognize that a nearby paragraph is a text equivalent for the image (when not marked up as such).
3. The checkpoint requires control of a content property that the user agent cannot recognize because of how the content has been encoded in a particular format. Some examples of this include:
    - captioning information that is "burned" into a video presentation and cannot be recognized as captions in the presentation format;
    - streamed content that cannot be fast forwarded or rewound;
    - information encoded in an unrecognized XML namespace;
    - information or relationships encoded in scripts in a manner that cannot be recognized. For instance, the requirements of checkpoint 3.3 would not apply for animation effects unrecognized in a script. Some input device behavior may be controlled by scripts in a manner that the user agent cannot recognize. For example, when the author uses event bubbling to dispatch events, the user agent is not likely to recognize the full set of elements that may receive those events; the user agent is expected to recognize which element has the explicitly associated event handler .

3.1.10 Example of building a conformance profile

The following example illustrates how to evaluate a user agent and build an appropriate conformance profile . This informative example does not illustrate a complete user agent evaluation.

Consider a user agent that:

- Supports keyboard and pointing device input, but does not support full operation through the pointing device. The user agent does not support voice input.
- Implements:
    - three formats that produce visually rendered text, T1, T2, and T3;

  - one audio format, A1;
  - two image formats, I1 and I2;
  - two video formats, V1 and V2, that are rendered by a plug-in;
  - two other animation formats (besides video, considered an animation format by definition).
- Does not support synthesized speech output.
- Implements functionalities to allow keyboard access to event handlers originally designed to be activated through a pointing device.
- Supports a selection mechanism.

Step 1: Identify a conformance level.

For this profile, we choose level Double-A. This establishes a set of requirements consisting of all of the requirements of all the priority 1 and 2 checkpoints.

Step 2: Identify content type labels.

For this profile, we must include VisualText since the user agent renders text visually.

For this profile, we also wish to include the labels Image, Video, and Audio, so the user agent must satisfy those requirements as well. Consider the following checkpoint:

4.4 Slow multimedia. (P1)

1. Allow the user to slow the presentation rate of rendered audio and animation content (including video and animated images).
2. As part of satisfying provision one of this checkpoint, for a visual track , provide at least one setting between 40% and 60% of the original speed.
3. As part of satisfying provision one of this checkpoint, for a prerecorded audio track including audio-only presentations , provide at least one setting between 75% and 80% of the original speed.
4. When the user agent allows the user to slow the visual track of a synchronized multimedia presentation to between 100% and 80% of its original speed, synchronize the visual and audio tracks (per checkpoint 2.6). Below 80%, the user agent is not required to render the audio track .

The second provision is specific to video, so must be satisfied for this profile. The third provision is specific to audio, so must be satisfied as well. The fourth provision involves synchronization, but the user agent does not implement any synchronized multimedia format (see step 6).

Note also the relevant normative exclusions for this checkpoint: the user agent is not required to satisfy the requirements of this checkpoint for audio and animations whose recognized role is to create a purely stylistic effect. In our example, the user agent provides the functionality for all audio and animations – even those used for purely stylistic effects – even though this is not required.

Although the user agent implements two animation formats, it only meets some, but not all, of the requirements associated with the Animation label. Therefore, we do not include it in the profile.

Note: A conformance claim will indicate that the plug-in renders the video.

Step 3: Identify support for event handlers

In this example, the user agent supports functionalities that promote input-device independent access to event handlers . Therefore, we can include the Events label in the profile.

Step 4: Identify support for the selection.

In this example, since the user agent implements a selection mechanism , the profile must include the Selection label (and the user agent must satisfy the associated requirements).

Step 5: Identify support for pointer and/or voice.

Since the user agent does not fully support operation through the pointing device alone or voice input alone, we exclude the Pointer and Voice labels from the profile.

Step 6: Identify requirements that do not apply.

In step 2 we saw that the fourth provision of checkpoint 4.4 did not apply since the user agent implements no synchronized multimedia format.

Other provisions that do not apply must also be documented.

Resulting profile

The profile resulting from these would include the following information:

> For "User Agent Accessibility Guidelines 1.0", http://www.w3.org/TR/2002/WD-UAAG10-20020821:
>
> - Conformance level: Double-A
> - Supported conformance profile labels: VisualText (T1, T2, T3), Image (I1, I2), Video (V1, V2), Audio (A1), Events, and Selection
> - Applicability: For checkpoint 4.4, provision three of checkpoint does not apply because the user agent does not implement any formats that support synchronized multimedia.

3.2 Conformance claims

A claim is **well-formed** if it meets the following two conditions.

Condition 1: The claim must include the following information:

1. The date of the claim.
2. The chosen conformance profile .
3. Information about the user agent. The user agent may consist of one or more

component. For each component, the claim must include the following:

- Name and version information for the component. Version information must be sufficient to identify the user agent (e.g., vendor name, version number, minor release number, required patches or updates, natural language of the user interface or documentation). The version information may refer to a range of user agents (e.g., "this claim refers to all user agents version 6.x").
- Name and version information for the operating environment (or environments) in which the component is running.

Condition 2: At least one version of the claim must conform to the "Web Content Accessibility Guidelines 1.0" *[WCAG10]* , level A. This claim may appear on the Web, on a CD-ROM, etc. If a conformance icon is part of a claim on the Web, it must link to the W3C explanation of the icon.

This specification imposes no restrictions on the format used to make a well-formed claim. For instance, the claim may be marked up using HTML (see sample claim), or expressed in the Resource Description Framework (RDF) *[RDF10]* .

Here is a sample conformance claim (expressed in HTML):

```
<p>On 21 August 2002, UserAgent X (version 2.3) running on
MyOperatingSystem (version 4.2) conforms to <abbr title="the
World Wide Web Consortium">W3C</abbr>'s "User Agent
Accessibility Guidelines 1.0",
http://www.w3.org/TR/2002/WD-UAAG10-20020821. Conformance
level: Double A. Supported conformance profile labels:
VisualText, Image, Animation, Audio, Events, and Selection. A
<a href="http://example.com/checkpoints">list</a> of formats
used to satisfy the requirements, and of checkpoints that do
not apply is available online. The specifications that are
part of this profile are W3C's HTML 4.0, CSS2, PNG, and SVG
(where each acronym links to the corresponding specification).
</p>
```

3.2.1 Validity of a claim

A conformance claim is valid if it is well-formed and if the user agent satisfies the requirements of the chosen conformance profile.

The document has been designed to help non-experts evaluate the validity of conformance claims. Some checkpoints may require interpretation and judgment. In some cases, although a requirement is clearly stated, without documentation or feedback from developers (e.g., about implemented APIs) it may be difficult to evaluate whether a user agent has satisfied the requirement. Some checkpoints (e.g., those requiring developers to follow conventions or implement specifications defined outside this document) are inherently more open to interpretation than others.

It is not currently possible to evaluate the validity of a claim automatically.

Note: The checklist *[UAAG10-CHECKLIST]* is designed to help people evaluate user agents. The User Agent Accessibility Guidelines Working Group makes available additional test suites, guides, and other tools to help people evaluate user agents for conformance.

3.2.2 Restricted functionality and valid claims

User agents do not conform to this document on a per-resource basis; claims are not as specific as "the user agent conforms for this particular Web page." A claim is valid if the user agent satisfies the requirements identified by the claim for most general-purpose content, in ordinary operating conditions.

In some cases, the author's content may limit the user agent's functionality for specific reasons, such as to protect intellectual property rights, to provide a read-only view (allowing no user interaction), or to limit interaction for a specialized purpose (e.g., a "written" driving test). Content that limits the functionality of the user agent in some cases does not automatically invalidate a claim about the user agent.

3.2.3 Responsibility for claims

A conformance claim (with or without an accompanying conformance icon) is an assertion that a user agent has satisfied the requirements of a chosen conformance profile. Claimants (or relevant assuring parties) are solely responsible for the validity of their claims, keeping claims up to date, and proper use of the conformance icons .

The existence of a conformance claim (with or without an accompanying conformance icon) does not imply that W3C has reviewed the claim or assured its validity. As of the publication of this document, W3C does not act as an assuring party, but it may do so in the future, or it may establish recommendations for assuring parties.

Claimants are expected to modify or retract a claim if it may be demonstrated that the claim is not valid. Claimants are encouraged to claim conformance to the most recent User Agent Accessibility Guidelines Recommendation available.

This specification imposes no restrictions about:

- who may make a claim (e.g., vendors about their own user agents, third parties about those user agents, journalists about user agents, etc.), *or*
- where claims may be published (e.g., on the Web or in paper documentation).

3.2.4 Conformance icons

People may use a conformance icon (or, "conformance logo") on a Web site, on user agent packaging, in documentation, etc. It is meaningless to use a conformance icon on its own, i.e., to use the icon without an associated well-formed claim .

Draft Note: *In the event this document becomes a W3C Recommendation this document will link to the W3C Web site for additional information about the icons and how to use them.*

3.3 UAAG 1.0 requirements in other specifications

Authors of technical specifications (such as W3C Recommendations) should incorporate the requirements of UAAG 1.0 as part of conformance to their specifications. This may be done by direct inclusion , or by reference using a conformance profile . Direct inclusion promotes the integration of specialized accessibility requirements; inclusion by reference is easier and less prone to error.

3.3.1 General advice

1. Identify accessibility features of the specification where they are defined (see checkpoint 8.1). Optionally, create an appendix of these accessibility features as well.
2. Remember to include user interface requirements as part of conformance to the specification. Authors of technical specifications tend to focus more on the rendering process or other content-related behavior, and less on user interface requirements. UAAG 1.0 makes a number of user interface requirements that authors will need to consider (such as those in guideline 5 pertaining to viewport behavior).
3. Include at least an informative reference to UAAG 1.0 and Techniques for UAAG 1.0. See the section on how to refer to UAAG 1.0 for more information.
4. Consult the User Agent Accessibility Guidelines Working Group when a question arises about how a checkpoint applies for a technology, such as whether a term is used differently between UAAG 1.0 and the technical specification.

For more information on designing specifications that promote accessibility, refer to W3C's "XML Accessibility Guidelines" *[XAG10]* .

3.3.2 Direct inclusion of requirements

1. Rather than including the generic UAAG 1.0 requirements, tailor them to the specification. Be specific in the requirements, and include (in context) a reference to the original UAAG 1.0 checkpoint. The following examples illustrate what is meant by direct inclusion:
   - In an HTML specification, where the `script`, `applet`, and `object` elements are defined, include a statement such as "Per checkpoint 3.4 of UAAG 1.0, a conforming user agent must allow configuration not to execute scripts, applets, or other executable content."
   - In a CSS specification, where the `'text-decoration'` property is defined, include a statement such as "A conforming user agent must either: (a) allow configuration to override the `'blink'` value with the 'none' value, or (b) ignore the `'blink'` value. This is required by checkpoint 3.3 of

UAAG 1.0 [UAAG10]."

Note how these examples refer to the specific elements, attributes, properties, etc., of the specifications.

2. It is better to include some UAAG 1.0 requirements in a specification than no UAAG 1.0 requirements. However, since UAAG 1.0 requirements are designed to complement one another, arbitrary selection of requirements may result in accessibility gaps. Authors should include requirements according to the groups defined by the conformance profile labels .

3.3.3 Inclusion by reference

Section G.5 of the SVG 1.0 Recommendation *[SVG]* states:

> Additionally, an authoring tool which is a Conforming SVG Generator conforms to all of the Priority 1 accessibility guidelines from the document "Authoring Tool Accessibility Guidelines 1.0" that are relevant to generators of SVG content.

This statement requires conformance to the Authoring Tool Accessibility Guidelines as part of conformance to SVG 1.0 (for certain classes of tools). This type of "conformance requirement by reference" is also possible for UAAG 1.0, by inclusion of a conformance profile .

The following is a (partial) example of a conformance profile for the MyFormat specification (expressed in plain text):

As part of conformance to MyFormat 1.0, a user agent must satisfy the following conformance profile:

- For "User Agent Accessibility Guidelines 1.0", http://www.w3.org/TR/2002/WD-UAAG10-20020821
- Conformance level A
- Content type labels: VisualText, Image, Animation, and Video. A conforming MyFormat user agent must satisfy the requirements associated with those labels.
- Selection: A conforming MyFormat user agent must implement a text selection mechanism, and therefore satisfy the requirements associated with the UAAG 1.0 selection label. A conforming MyFormat user agent is only required to allow users to select text content.
- Applicability: The following UAAG 1.0 checkpoints do not apply to MyFormat and therefore do not need to be satisfied for conformance to this specification:
  - 1.2, 3.4, 9.5, 9.6: MyFormat does not allow inclusion of scripts. Thus, there are no author-supplied event handlers.
  - 2.4, 2.6: MyFormat does not involve synchronization.
  - 2.5, 4.6: MyFormat does not define captions.
  - 10.1: MyFormat does not define tables.
  - (And so on)
- Implemented specification: MyFormat 1.0, 1 January 2002 draft, available at

http://www.example.com/MyFormat/.

See the section on how to refer to UAAG 1.0 for what should appear in the references section of the specification.

4. Glossary

This glossary is normative . However, some terms (or parts of explanations of terms) may not have an impact on conformance.

Note: In this document, glossary terms generally link to the corresponding entries in this section. These terms are also highlighted through style sheets and identified as glossary terms through markup.

Activate
: In this document, the verb "to activate" means (depending on context) either:

- To execute or carry out one or more behaviors associated with an enabled element .
- To execute or carry out one or more behaviors associated with a component of the user agent user interface .

The effect of activation depends on the type of the user interface control . For instance, when a link is activated, the user agent generally retrieves the linked Web resource . When a form element is activated, it may change state (e.g., check boxes) or may take user input (e.g., a text entry field).

Alert
: In this document, "to alert" means to make the user aware of some event, without requiring acknowledgement. For example, the user agent may alert the user that new content is available on the server by displaying a text message in the user agent's status bar. See checkpoint 1.3 for requirements about alerts.

Animation
: In this document, an "animation" refers to content that, when rendered, creates a visual movement effect automatically (i.e., without manual user interaction). This definition of animation includes video and animated images. Animation techniques include:

- graphically displaying a sequence of snapshots within the same region (e.g., as is done for video and animated images). The series of snapshots may be provided by a single resource (e.g., an animated GIF image) or from distinct resources (e.g., a series of images downloaded continuously by the user agent).
- scrolling text (e.g., achieved through markup or style sheets).
- displacing graphical objects around the viewport (e.g., a picture of a ball that is moved around the viewport giving the impression that it is bouncing off of the viewport edges). For instance, the SMIL 2.0 *[SMIL20]* animation modules explain how to create such animation effects in a declarative manner (i.e., not by composition of successive snapshots).

Applet
: An applet is a program (generally written in the Java programming language) that is part of content , and that the user agent executes.

Application Programming Interface (API), ***conventional input/output/device API***
: An application programming interface (API) defines how communication may take place between applications.

Implementing APIs that are independent of a particular operating environment (as are the W3C DOM Level 2 specifications) may reduce implementation costs for multi-platform user agents and promote the development of multi-platform assistive technologies. Implementing conventional APIs for a particular operating environment may reduce implementation costs for assistive technology developers who wish to interoperate with more than one piece of software running on that operating environment.

A "device API" defines how communication may take place with an input or output device such as a keyboard, mouse, video card, etc.

In this document, an "input/output API" defines how applications or devices communicate with a user agent. As used in this document, input and output APIs include, but are not limited to, device APIs. Input and output APIs also include more abstract communication interfaces than those specified by device APIs. A "conventional input/output API" is one that is expected to be implemented by software running on a particular operating environment. For example, on desktop computers today, the conventional input APIs are for the mouse and keyboard. For touch screen devices or mobile devices, conventional input APIs may include stylus, buttons, voice, etc. The graphical display and sound card are considered conventional output devices for a graphical desktop computer environment, and each has an associated API.

Assistive technology

In the context of this document, an assistive technology is a user agent that:

1. relies on services (such as retrieving Web resources , parsing markup, etc.) provided by one or more other "host" user agents. Assistive technologies communicate data and messages with host user agents by using and monitoring APIs .
2. provides services beyond those offered by the host user agents to meet the requirements of users with disabilities. Additional services include alternative renderings (e.g., as synthesized speech or magnified content), alternative input methods (e.g., voice), additional navigation or orientation mechanisms, content transformations (e.g., to make tables more accessible), etc.

For example, screen reader software is an assistive technology because it relies on browsers or other software to enable Web access, particularly for people with visual and learning disabilities.

Examples of assistive technologies that are important in the context of this document include the following:

- screen magnifiers, which are used by people with visual disabilities to enlarge and change colors on the screen to improve the visual readability of rendered text and images.
- screen readers, which are used by people who are blind or have reading disabilities to read textual information through synthesized speech or braille displays.
- voice recognition software, which may be used by people who have some

physical disabilities.

- alternative keyboards, which are used by people with certain physical disabilities to simulate the keyboard.
- alternative pointing devices, which are used by people with certain physical disabilities to simulate mouse pointing and button activations.

Beyond this document, assistive technologies consist of software or hardware that has been specifically designed to assist people with disabilities in carrying out daily activities, e.g., wheelchairs, reading machines, devices for grasping, text telephones, vibrating pagers, etc. For example, the following very general definition of "assistive technology device" comes from the (U.S.) Assistive Technology Act of 1998 *[AT1998]* :

> Any item, piece of equipment, or product system, whether acquired commercially, modified, or customized, that is used to increase, maintain, or improve functional capabilities of individuals with disabilities.

Attribute

This document uses the term "attribute" in the XML sense: an element may have a set of attribute specifications (refer to the XML 1.0 specification *[XML]* section 3).

Audio

In this document, the term "audio" refers to content that encodes prerecorded sound.

Audio-only presentation

An audio-only presentation is content consisting exclusively of one or more audio tracks presented concurrently or in series. Examples of an audio-only presentation include a musical performance, a radio-style news broadcast, and a narration.

Audio track

An audio object is content rendered as sound through an audio viewport . An audio track is an audio object that is intended as a whole or partial presentation. An audio track may, but is not required to, correspond to a single audio channel (left or right audio channel).

Audio description

An audio description (called an "auditory description" in the Web Content Accessibility Guidelines 1.0 *[WCAG10]*) is either a prerecorded human voice or a synthesized voice (recorded or generated dynamically) describing the key visual elements of a movie or other animation. The audio description is synchronized with (and possibly included as part of) the audio track of the presentation, usually during natural pauses in the audio track . Audio descriptions include information about actions, body language, graphics, and scene changes.

Author styles

Authors styles are style property values that come from content (e.g., style sheets within a document, that are associated with a document, or that are generated by a server).

Captions

Captions are text transcripts that are synchronized with other audio tracks or visual tracks . Captions convey information about spoken words and non-spoken sounds such as sound effects. They benefit people who are deaf or hard-of-hearing, and anyone who cannot hear the audio (e.g., someone in a noisy environment). Captions are generally rendered graphically superimposed ("on top of") the synchronized visual track.

The term "open captions" generally refers to captions that are always rendered with a visual track; they cannot be turned off. The term "closed captions" generally refers to captions that may be turned on and off. The captions requirements of this document assume that the user agent can recognize the captions as such; see the section on applicability for more information.

Note: Other terms that include the word "caption" may have different meanings in this document. For instance, a "table caption" is a title for the table, often positioned graphically above or below the table. In this document, the intended meaning of "caption" will be clear from context.

Character encoding

A "character encoding" is a mapping from a character set definition to the actual code units used to represent the data. Refer to the Unicode specification *[UNICODE]* for more information about character encodings. Refer to "Character Model for the World Wide Web" *[CHARMOD]* for additional information about characters and character encodings.

Collated text transcript

A collated text transcript is a text equivalent of a movie or other animation. More specifically, it is the combination of the text transcript of the audio track and the text equivalent of the visual track . For example, a collated text transcript typically includes segments of spoken dialogue interspersed with text descriptions of the key visual elements of a presentation (actions, body language, graphics, and scene changes). See also the definitions of text transcript and audio description . Collated text transcripts are essential for individuals who are deaf-blind.

Conditional content

Conditional content is content that, by format specification, should be made available to users through the user interface, generally under certain conditions (e.g., based on user preferences or operating environment limitations). Some examples of conditional content mechanisms include:

- The "`alt`" attribute of the `IMG` element in HTML 4. According to section 13.2 of the HTML 4 specification (*[HTML4]*): "User agents must render alternate text when they cannot support images, they cannot support a certain image type or when they are configured not to display images.
- `OBJECT` elements in HTML 4. Section 13.3.1 of the HTML 4 specification (*[HTML4]*) explains the conditional rendering rules of (nested) `OBJECT` elements. The rules select among ordered alternatives according to user preferences or error conditions.

- The `switch` element and test attributes in SMIL 1.0. Sections 4.3 and 4.4, respectively, of SMIL 1.0 *[SMIL]* explain the conditional rendering rules of these features.
- SVG 1.0 *[SVG]* also includes a `switch` element and several attributes for conditional processing.
- The `NOSCRIPT` and `NOFRAMES` elements in HTML 4 *[HTML4]* allow the author to provide content under conditions when the user agent does not support scripts or frames, or the user has turned off support for scripts or frames.

Specifications vary in how completely they define how and when to render conditional content. For instance, the HTML 4 specification includes the rendering conditions for the "`alt`" attribute, but not for the "`title`" attribute. The HTML 4 specification does indicate that the "`title`" attribute should be available to users through the user interface ("Values of the title attribute may be rendered by user agents in a variety of ways...").

Note: The Web Content Accessibility Guidelines 1.0 requires that authors provide text equivalents for non-text content. This is generally done by using the conditional content mechanisms of a markup language. Since conditional content may not be rendered by default, the current document requires the user agent to provide access to unrendered conditional content (checkpoint 2.3 and checkpoint 2.9) as it may have been provided to promote accessibility.

Configure*, *control

In the context of this document, the verbs "to control" and "to configure" share in common the idea of governance such as a user may exercise over interface layout, user agent behavior, rendering style, and other parameters required by this document. Generally, the difference in the terms centers on the idea of *persistence*. When a user makes a change by "controlling" a setting, that change usually does not persist beyond that user session. On the other hand, when a user "configures" a setting, that setting typically persists into later user sessions. Furthermore, the term "control" typically means that the change can be made easily (such as through a keyboard shortcut) and that the results of the change occur immediately. The term "configure" typically means that making the change requires more time and effort (such as making the change via a series of menus leading to a dialog box, via style sheets or scripts, etc.). The results of "configuration" might not take effect immediately (e.g., due to time spent reinitializing the system, initiating a new session, rebooting the system).

In order to be able to configure and control the user agent, the user needs to be able to "write" as well as "read" values for these parameters. Configuration settings may be stored in a profile . The range and granularity of the changes that can be controlled or configured by the user may depend on limitations of the operating environment or hardware.

Both configuration and control may apply at different "levels": across Web resources (i.e., at the user agent level, or inherited from the operating environment), to the entirety of a Web resource, or to components of a Web resource (e.g., on a per-element basis).

A ***global configuration*** is one that applies across elements of the same Web resource, as well as across Web resources. A global configuration may be implemented by more than one setting (e.g., per component of the user agent). For instance, when a user agent consists of a browser that renders HTML and a plug-in that renders SVG, to satisfy the global configuration requirements of this document, the browser may provide one setting and the plug-in another.

User agents may allow users to choose configurations based on various parameters, such as hardware capabilities, natural language, etc.

Note: In this document, the noun "control" refers to a user interface control .

Content

In this specification, the noun "content" is used in three ways:

1. It is used to mean the document object as a whole or in parts.
2. It is used to mean the content of an HTML or XML element, in the sense employed by the XML 1.0 specification (*[XML]* , section 3.1): "The text between the start-tag and end-tag is called the element's content." Context should indicate that the term content is being used in this sense.
3. It is used in the terms non-text content and text content .

Empty content (which may be conditional content) is either a null value or an empty string (i.e., one that is zero characters long). For instance, in HTML, `alt=""` sets the value of the "`alt`" attribute to the empty string. In some markup languages, an element may have empty content (e.g., the `HR` element in HTML).

Device-independence

Device-independence refers to the ability to make use of software with any appropriate supported input or output device.

Document object, ***Document Object Model*** (DOM)

In general usage, the term "document object" refers to the user agent's representation of data (e.g., a document). This data generally comes from the document source , but may also be generated (from style sheets, scripts, transformations, etc.), produced as a result of preferences set within the user agent, added as the result of a repair performed automatically by the user agent, etc. Some data that is part of the document object is routinely rendered (e.g., in HTML, what appears between the start and end tags of elements and the values of attributes such as "alt", "title", and "summary"). Other parts of the document object are generally processed by the user agent without user awareness, such as DTD- or schema-defined names of element types and attributes, and other attribute values such as "href", "id", etc. These guidelines require that users have access to both kinds of data through the user interface. Most of the requirements of this document apply to the document object after its construction. However, a few checkpoints (e.g., checkpoint 2.7 and checkpoint 2.10) may affect the construction of the document object.

A "document object model" is the abstraction that governs the construction of the user agent's document object. The document object model employed by different user agents may vary in implementation and sometimes in scope. This specification requires that user agents implement the APIs defined in Document Object Model (DOM) Level 2 specifications (*[DOM2CORE]* and *[DOM2STYLE]*) for access to HTML, XML, and CSS content. These DOM APIs allow authors to access and modify the content via a scripting language (e.g., JavaScript) in a consistent manner across different scripting languages. As a standard interface, the DOM APIs make it easier not just for authors, but for assistive technology developers to extract information and render it in ways most suited to the needs of particular users.

Document character set

A document character set (a concept taken from SGML) is a sequence of abstract characters that may appear in Web content represented in a particular format (such as HTML, XML, etc.). A document character set consists of:

- A "repertoire": A set of abstract characters, such as the Latin letter "A", the Cyrillic letter "I", the Chinese character meaning "water", etc.
- Code positions: A set of integer references to characters in the repertoire.

For instance, the character set required by the HTML 4 specification *[HTML4]* is defined in the Unicode specification *[UNICODE]* . Refer to "Character Model for the World Wide Web" *[CHARMOD]* for more information about document character sets.

Document source, ***text source***

In this document, the term "document source" refers to the data that the user agent receives as the direct result of a request for a Web resource (e.g., as the result of an HTTP/1.1 *[RFC2616]* "GET", or as the result of viewing a resource on the local file system). The document source generally refers to the "payload" of the user agent's request, and does not generally include information exchanged as part of the transfer protocol. The document source is data that is prior to any repair by the user agent (e.g., prior to repairing invalid markup). "Text source" refers to document source that is composed of text .

Documentation

Documentation refers to information that supports the use of a user agent. This information may be found in manuals, installation instructions, the help system, tutorials, etc. Documentation may be distributed (e.g., some parts may be delivered on CD-ROM, others on the Web). Refer to guideline 12 for information about documentation requirements.

Element, ***element type***

This document uses the terms "element" and "element type" in the sense employed by the XML 1.0 specification (*[XML]* , section 3): an element type is a syntactic construct of a document type definition (DTD) for its application. This sense is also relevant to structures defined by XML schemas. The document also uses the term "element" more generally to mean a type of content (such as video or sound) or a logical construct (such as a header or list).

Enabled element*, *disabled element

An enabled element is a piece of content with associated behaviors that may be activated through the user interface or through an API . The set of elements that a user agent enables is generally derived from, but is not limited to, the set of interactive elements defined by implemented markup languages.

Some elements may only be enabled elements for part of a user session. For instance, an element may be disabled by a script as the result of user interaction. Or, an element may only be enabled during a given time period (e.g., during part of a SMIL 1.0 *[SMIL]* presentation). Or, the user may be viewing content in "read-only" mode, which may disable some elements.

A disabled element is a piece of content that is potentially an enabled element, but is not in the current session. One example of a disabled element is a menu item that is unavailable in the current session; it might be "greyed out" to show that it is disabled. Generally, disabled elements will be interactive elements that are not enabled in the current session. This document distinguishes disabled elements (not currently enabled) from non-interactive elements (never enabled).

For the requirements of this document, user selection does not constitute user interaction with enabled elements. See the definition of content focus .

Note: Enabled and disabled elements come from content; they are not part of the user agent user interface .

Note: The term "active element" is not used in this document since it may suggest several different concepts, including: interactive element, enabled element, an element "in the process of being activated" (which is the meaning of ':active' in CSS2 *[CSS2]* , for example).

Equivalent (for content)

The term "equivalent" is used in this document as it is used in the Web Content Accessibility Guidelines 1.0 *[WCAG10]* :

> Content is "equivalent" to other content when both fulfill essentially the same function or purpose upon presentation to the user. In the context of this document, the equivalent must fulfill essentially the same function for the person with a disability (at least insofar as is feasible, given the nature of the disability and the state of technology), as the primary content does for the person without any disability.

Equivalents include text equivalents (e.g., text equivalents for images, text transcripts for audio tracks, or collated text transcripts for a movie) and non-text equivalents (e.g., a prerecorded audio description of a visual track of a movie, or a sign language video rendition of a written text).

Each markup language defines its own mechanisms for specifying conditional content , and these mechanisms may be used by authors to provide text equivalents. For instance, in HTML 4 *[HTML4]* or SMIL 1.0 *[SMIL]* , authors may use the "`alt`" attribute to specify a text equivalent for some elements. In

HTML 4, authors may provide equivalents and other conditional content in attribute values (e.g., the "summary" attribute for the `TABLE` element), in element content (e.g., `OBJECT` for external content it specifies, `NOFRAMES` for frame equivalents, and `NOSCRIPT` for script equivalents), and in prose. Please consult the Web Content Accessibility Guidelines 1.0 *[WCAG10]* and its associated Techniques document *[WCAG10-TECHS]* for more information about equivalents.

Events and scripting, event handler, event type

User agents often perform a task when an event having a particular "event type" occurs, including user interface, changes to content, loading of content, a request from the operating environment , etc. Some markup languages allow authors to specify that a script, called an ***event handler***, be executed when an event of a given type occurs. An event handler is ***explicitly associated with an element*** when the event handler is associated with that element through markup or the DOM. The term "event bubbling" describes a programming style where a single event handler dispatches events to more than one element. In this case, the event handlers are not explicitly associated with the elements receiving the events (except for the single element that dispatches the events).

Note: The combination of HTML, style sheets, the Document Object Model (DOM), and scripting is commonly referred to as "Dynamic HTML" or DHTML. However, as there is no W3C specification that formally defines DHTML, this document only refers to event handlers and scripts.

Explicit user request

In this document, the term "explicit user request" refers to any user interaction through the user agent user interface (not through rendered content), the focus , or the selection . User requests are made, for example, through user agent user interface controls and keyboard bindings.

Some examples of explicit user requests include when the user selects "New viewport", responds "Yes" to a prompt in the user agent’s user interface, configures the user agent to behave in a certain way, or changes the selection or focus with the keyboard or pointing device.

Note: Users make mistakes. For example, a user may inadvertently respond "yes" to a prompt when they meant "no." In this document, this type of mistake is still considered an explicit user request.

Focus, ***content focus***, ***user interface focus***, ***current focus***

In this document, the term "content focus" (required by checkpoint 9.1) refers to a user agent mechanism that has all of the following properties:

1. It designates zero or one element in content that is either enabled or disabled . In general, the focus should only designate enabled elements, but it may also designate disabled elements.
2. It has state: the user may "set it" (programmatically or through the user interface) on an enabled element. Events may be triggered when the focus is set (or unset). Which events are triggered depends on the content (e.g., HTML events and CSS pseudo-classes) or user interface settings.

3. Once it has been set, it may be used to trigger other behaviors associated with the enabled element (e.g., the user may activate a link or change the state of a form control). These behaviors may be triggered programmatically or through the user interface (e.g., through keyboard events).

User interface mechanisms may resemble content focus, but do not satisfy all of the properties. For example, text editors often implement a "caret" that indicates the current location of text input or editing. The caret may have state and may respond to input device events, but it does not enable users to activate the behaviors associated with enabled elements.

The user interface focus shares the properties of the content focus except that, rather than designating pieces of content, it designates zero or one control of the user agent user interface that has associated behaviors (e.g., radio button, text box, menu, etc.).

On the screen, the content focus may be highlighted using colors, fonts, graphics, magnification, etc. The content focus may also be highlighted when rendered as synthesized speech, for example through changes in speech prosody. The dimensions of the rendered content focus may exceed those of the viewport.

In this document, each viewport is expected to have at most one content focus and at most one user interface focus. This document includes requirements for content focus only, for user interface focus only, and for both. When a requirement refers to both, the term "focus" is used.

When several viewports coexist, at most one viewport's content focus **or** user interface focus responds to input events; this is called the current focus.

Graphical
In this document, the term "graphical" refers to information (text, colors, graphics, images, animations, etc.) rendered for visual consumption.

Highlight
In this document, "to highlight" means to emphasize through the user interface. For example, user agents highlight which content is selected or focused. Graphical highlight mechanisms include dotted boxes, underlining, and reverse video. Synthesized speech highlight mechanisms include alterations of voice pitch and volume ("speech prosody").

Image
This document uses the term "image" to refer (as is commonly the case) to pictorial content . However, in this document, term image is limited to static (i.e., unmoving) visual information. See also the definition of animation .

Input configuration
An input configuration is the set of "bindings" between user agent functionalities and user interface input mechanisms (e.g., menus, buttons, keyboard keys, voice commands, etc.). The default input configuration is the set of bindings the user finds after installation of the software; it must be documented (per checkpoint 12.3]). Input configurations may be affected by author-specified

bindings (e.g., through the "accesskey" attribute of HTML 4 *[HTML4]*).

Interactive element, ***non-interactive element***,
: An interactive element is piece of content that, by specification, may have associated behaviors to be executed or carried out as a result of user or programmatic interaction. For instance, the interactive elements of HTML 4 *[HTML4]* include: links, image maps, form elements, elements with a value for the "longdesc" attribute, and elements with event handlers explicitly associated with them (e.g., through the various "on" attributes). The role of an element as an interactive element is subject to applicability . A non-interactive element is an element that, by format specification, does not have associated behaviors. The expectation of this document is that interactive elements become enabled elements in some sessions, and non-interactive elements never become enabled elements.

Natural language
: Natural language is spoken, written, or signed human language such as French, Japanese, and American Sign Language. On the Web, the natural language of content may be specified by markup or HTTP headers. Some examples include the "lang" attribute in HTML 4 (*[HTML4]* section 8.1), the "xml:lang" attribute in XML 1.0 (*[XML]* , section 2.12), the HTML 4 "hreflang" attribute for links in HTML 4 (*[HTML4]* , section 12.1.5), the HTTP Content-Language header (*[RFC2616]* , section 14.12) and the Accept-Language request header (*[RFC2616]* , section 14.4). See also the definition of script .

Normative, ***informative***
: What is identified as "normative" is required for conformance (noting that one may conform in a variety of well-defined ways to this document). What is identified as "informative" (sometimes, "non-normative") is never required for conformance.

Operating environment
: The term "operating environment" refers to the environment that governs the user agent's operation, whether it is an operating system or a programming language environment such as Java.

Override
: In this document, the term "override" means that one configuration or behavior preference prevails over another. Generally, the requirements of this document involve user preferences prevailing over author preferences and user agent default settings and behaviors. Preferences may be multi-valued in general (e.g., the user prefers blue over red or yellow), and include the special case of two values (e.g., turn on or off blinking text content).

Placeholder
: A placeholder is content generated by the user agent to replace author-supplied content. A placeholder may be generated as the result of a user preference (e.g., to not render images) or as repair content (e.g., when an image cannot be found). Placeholders can be any type of content, including text, images, and audio cues.

This document includes requirements that the user be able to view the original author-supplied content associated with a placeholder. To satisfy these requirements, the user agent might render the content in place of the placeholder or in a separate viewport (leaving the placeholder as is). A request to view the original content associated with a placeholder is considered an explicit user request to render that content.

This document does not require user agents to include placeholders in the document object . A placeholder that is inserted in the document object should conform to the Web Content Accessibility Guidelines 1.0 *[WCAG10]* . If a placeholder is not part of the document object, it is part of the user interface only (and subject, for example, to checkpoint 1.3).

Plug-in

A plug-in is a program that runs as part of the user agent and that is *not* part of content . Users generally choose to include or exclude plug-ins from their user agent.

Point of regard

The point of regard is a position in rendered content that the user is presumed to be viewing. The dimensions of the point of regard may vary. For example, it may be a point (e.g., a moment in an audio rendering or a cursor in a graphical rendering), or a range of text (e.g., focused text), or a two-dimensional area (e.g., content rendered through a two-dimensional graphical viewport). The point of regard is almost always within the viewport, but it may exceed the spatial or temporal dimensions of the viewport (see the definition of rendered content for more information about viewport dimensions). The point of regard may also refer to a particular moment in time for content that changes over time (e.g., an audio-only presentation). User agents may determine the point of regard in a number of ways, including based on viewport position in content, content focus , selection , etc. A user agent should not change the point of regard unexpectedly as this may disorient the user. The point of regard should be available programmatically (e.g., for assistive technologies).

Profile

A profile is a named and persistent representation of user preferences that may be used to configure a user agent. Preferences include input configurations, style preferences, natural language preferences, etc. In operating environments with distinct user accounts, profiles enable users to reconfigure software quickly when they log on, and profiles may be shared by several users. Platform-independent profiles are useful for those who use the same user agent on different platforms.

Prompt

In this document, "to prompt" means to require input from the user. The user agent should allow users to configure how they wish to be prompted. For instance, for a user agent functionality X, configurations might include: always prompt me before doing X, always do X without prompting me, never do X but tell me when you could have, never do X and never tell me that you could have, etc.

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, on paper, through loudspeakers, on a braille display, on a mobile device, etc. Style information (e.g., fonts, colors, synthesized speech prosody, etc.) may come from the elements themselves (e.g., certain font and phrase elements 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. Generally in this document, the term "property" has the meaning defined in CSS 2 (*[CSS2]* , section 3). 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 installed is called the property's default value.

Recognize

Authors encode information in markup languages, style sheet languages, scripting languages, protocols, etc. When the information is encoded in a manner that allows the user agent to process it with certainty, the user agent can "recognize" the information. For instance, HTML allows authors to specify a heading with the H1 element, so a user agent that implements HTML can recognize that content as a heading. If the author creates headings using a visual effect alone (e.g., by increasing the font size), then the author has encoded the heading in a manner that does not allow the user agent to recognize it as a heading.

Some requirements of this document depend on content roles, content relationships, timing relationships, and other information supplied by the author. These requirements only apply when the author has encoded that information in a manner that the user agent can recognize. See the section on conformance for more information about applicability.

In practice, user agents will rely heavily on information that the author has encoded in a markup language or style sheet language. On the other hand, behaviors, style, meaning encoded in a script , and markup in an unfamiliar XML namespace may not be recognized by the user agent as easily or at all. The Techniques document *[UAAG10-TECHS]* lists some markup known to affect accessibility that user agents can recognize.

Rendered content, ***rendered text***

Rendered content is the part of content that the user agent makes available to the user's senses of sight and hearing (and only those senses for the purposes of this document). Any content that causes an effect that may be perceived through these senses constitutes rendered content. This includes text characters, images, style sheets, scripts, and anything else in content that, once processed, may be perceived through sight and hearing.

The term "rendered text" refers to text content that is rendered in a way that communicates information about the characters themselves, whether visually or as synthesized speech.

In the context of this document, ***invisible content*** is content that influences graphical rendering of other content but is not rendered itself. Similarly, ***silent content*** is content that influences audio rendering of other content but is not rendered itself. Neither invisible nor silent content is considered rendered content.

Repair content, ***repair text***

In this document, the term "repair content" refers to content generated by the user agent in order to correct an error condition. "Repair text" means repair content consisting only of text . Some error conditions that may lead to the generation of repair content include:

- Erroneous or incomplete content (e.g., ill-formed markup, invalid markup, missing conditional content that is required by format specification);
- Missing resources for handling or rendering content (e.g., the user agent lacks a font family to display some characters, the user agent does not implement a particular scripting language).

This document does not require user agents to include repair content in the document object . Repair content inserted in the document object should conform to the Web Content Accessibility Guidelines 1.0 *[WCAG10]* . For more information about repair techniques for Web content and software, refer to "Techniques for Authoring Tool Accessibility Guidelines 1.0" *[ATAG10-TECHS]* .

Script

In this document, the term "script" almost always refers to a scripting (programming) language used to create dynamic Web content. However, in checkpoints referring to the written (natural) language of content, the term "script" is used as in Unicode *[UNICODE]* to mean "A collection of symbols used to represent textual information in one or more writing systems."

Information encoded in scripts may be difficult for a user agent to recognize . For instance, a user agent is not expected to recognize that, when executed, a script will calculate a factorial. The user agent will be able to recognize some information in a script by virtue of implementing the scripting language or a known program library (e.g., the user agent is expected to recognize when a script will open a viewport or retrieve a resource from the Web).

Selection, ***current selection***

In this document, the term "selection" refers to a user agent mechanism for identifying a range of content (e.g., text, images, etc.). Generally, user agents limit selection to text content (e.g., one or more fragments of text). The selection may be structured (based on the document tree) or unstructured (e.g., text-based). The range may be empty.

On the screen, the selection may be highlighted using colors, fonts, graphics, magnification, etc. The selection may also be highlighted when rendered as synthesized speech, for example through changes in speech prosody. The dimensions of the rendered selection may exceed those of the viewport.

The selection may be used for a variety of purposes: for cut and paste operations, to designate a specific element in a document for the purposes of a query, as an indication of point of regard , etc.

The selection has state, and the user may "set it" (programmatically or through the user interface).

In this document, each viewport is expected to have at most one selection. When several viewports coexist, at most one viewport's selection responds to input events; this is called the current selection.

See the section on the Selection label for information about implementing a selection and conformance .

Note: Some user agents may also implement a selection for designating a range of information in the user agent user interface . The current document only includes requirements for a content selection mechanism.

Serial access, ***sequential navigation***

In this document, the expression "serial access" refers to one-dimensional access to rendered content. Some examples of serial access include listening to an audio stream or watching a video (both of which involve one temporal dimension), or reading a series of lines of braille one line at a time (one spatial dimension). Many users with blindness have serial access to content rendered as audio, synthesized speech, or lines of braille.

The expression "sequential navigation" refers to navigation through an ordered set of items (e.g., the enabled elements in a document, a sequence of lines or pages, or a sequence of menu options). Sequential navigation implies that the user cannot skip directly from one member of the set to another, in contrast to direct or structured navigation (see guideline 9 for information about these types of navigation). Users with blindness or some users with a physical disability may navigate content sequentially (e.g., by navigating through links, one by one, in a graphical viewport with or without the aid of an assistive technology). Sequential navigation is important to users who cannot scan rendered content visually for context and also benefits users unfamiliar with content. The increments of sequential navigation may be determined by a number of factors, including element type (e.g., links only), content structure (e.g., navigation from heading to heading), and the current navigation context (e.g., having navigated to a table, allow navigation among the table cells).

Users with serial access to content or who navigate sequentially may require more time to access content than users who use direct or structured navigation.

Support, ***implement***, ***conform***

In this document, the terms "support", "implement", and "conform" all refer to what a developer has designed a user agent to do, but they represent different degrees of specificity. A user agent "supports" general classes of objects, such as "images" or "Japanese". A user agent "implements" a specification (e.g., the PNG and SVG image format specifications, a particular scripting language, etc.), or an API (e.g., the DOM API) when it has been programmed to follow all or part of a specification. A user agent "conforms to" a specification when it implements the specification *and* satisfies its conformance criteria. This document includes some conformance requirements to other specifications (e.g., to a particular level of the "Web Content Accessibility Guidelines 1.0" *[WCAG10]*).

Synchronize

In this document, "to synchronize" refers to the act of time-coordinating two or more presentation components (e.g., in a multimedia presentation, a visual track with captions). For Web content developers, the requirement to synchronize means to provide the data that will permit sensible time-coordinated rendering by a user agent. For example, Web content developers can ensure that the segments of caption text are neither too long nor too short, and that they map to segments of the visual track that are appropriate in length. For user agent developers, the requirement to synchronize means to present the content in a sensible time-coordinated fashion under a wide range of circumstances including technology constraints (e.g., small text-only displays), user limitations (slow reading speeds, large font sizes, high need for review or repeat functions), and content that is sub-optimal in terms of accessibility.

Text

In this document, the term "text" used by itself refers to a sequence of characters from a markup language's document character set . Refer to the "Character Model for the World Wide Web " *[CHARMOD]* for more information about text and characters. **Note:** This document makes use of other terms that include the word "text" that have highly specialized meanings: collated text transcript , non-text content , text content , non-text element , text element , text equivalent , and text transcript .

Text content, ***non-text content***, ***text element***, ***non-text element***, ***text equivalent***, ***non-text equivalent***

As used in this document a "text element" adds text characters to either content or the user interface . Both in the Web Content Accessibility Guidelines 1.0 *[WCAG10]* and in this document, text elements are presumed to produce text that can be understood when rendered visually, as synthesized speech, or as Braille. Such text elements benefit at least these three groups of users:

1. visually-displayed text benefits users who are deaf and adept in reading visually-displayed text;
2. synthesized speech benefits users who are blind and adept in use of synthesized speech;
3. braille benefits users who are blind, and possibly deaf-blind, and adept at reading braille.

A text element may consist of both text and non-text data. For instance, a text element may contain markup for style (e.g., font size or color), structure (e.g., heading levels), and other semantics. The essential function of the text element should be retained even if style information happens to be lost in rendering.

A user agent may have to process a text element in order to have access to the text characters. For instance, a text element may consist of markup, it may be encrypted or compressed, or it may include embedded text in a binary format (e.g., JPEG).

"Text content" is content that is composed of one or more text elements. A "text equivalent" (whether in content or the user interface) is an equivalent composed of one or more text elements. Authors generally provide text equivalents for content by using the conditional content mechanisms of a specification.

A "non-text element" is an element (in content or the user interface) that does not have the qualities of a text element. "Non-text content" is composed of one or more non-text elements. A "non-text equivalent" (whether in content or the user interface) is an equivalent composed of one or more non-text elements.

Text decoration

In this document, a "text decoration" is any stylistic effect that the user agent may apply to visually rendered text that does not affect the layout of the document (i.e., does not require reformatting when applied or removed). Text decoration mechanisms include underline, overline, and strike-through.

Text transcript

A text transcript is a text equivalent of audio information (e.g., an audio-only presentation or the audio track of a movie or other animation). It provides text for both spoken words and non-spoken sounds such as sound effects. Text transcripts make audio information accessible to people who have hearing disabilities and to people who cannot play the audio. Text transcripts are usually created by hand but may be generated on the fly (e.g., by voice-to-text converters). See also the definitions of captions and collated text transcripts .

User agent

In this document, the term "user agent" is used in two ways:

1. The software and documentation components that together, conform to the requirements of this document. This is the most common use of the term in this document and is the usage in the checkpoints.
2. Any software that retrieves and renders Web content for users. This may include Web browsers, media players, plug-ins , and other programs – including assistive technologies -- that help in retrieving and rendering Web content.

User agent default styles

User agent default styles are style property values applied in the absence of any author or user styles. Some markup languages specify a default rendering for documents in that markup language. Other specifications may not specify default styles. For example, XML 1.0 *[XML]* does not specify default styles for XML documents. HTML 4 *[HTML4]* does not specify default styles for HTML documents, but the CSS 2 *[CSS2]* specification suggests a sample default style sheet for HTML 4 based on current practice.

User interface, ***user interface control***

For the purposes of this document, user interface includes both:

1. the ***user agent user interface***, i.e., the controls (e.g., menus, buttons, prompts, and other components for input and output) and mechanisms (e.g., selection and focus) provided by the user agent ("out of the box") that are not created by content .
2. the "content user interface", i.e., the enabled elements that are part of content, such as form controls, links, applets , etc.

The document distinguishes them only where required for clarity. For more information, see the section on requirements for content, for user agent features, or both .

The term "user interface control" refers to a component of the user agent user interface or the content user interface, distinguished where necessary.

User styles

User styles are style property values that come from user interface settings, user style sheets, or other user interactions.

Views, ***viewports***

The user agent renders content through one or more viewports. Viewports include windows, frames, pieces of paper, loudspeakers, virtual magnifying glasses, etc. A viewport may contain another viewport (e.g., nested frames). User agent user interface controls such as prompts, menus, alerts, etc., are not viewports.

Graphical and tactile viewports have two spatial **dimensions**. A viewport may also have temporal dimensions, for instance when audio, speech, animations, and movies are rendered. When the dimensions (spatial or temporal) of rendered content exceed the dimensions of the viewport, the user agent provides mechanisms such as scroll bars and advance and rewind controls so that the user can access the rendered content "outside" the viewport. Examples include: when the user can only view a portion of a large document through a small graphical viewport, or when audio content has already been played.

When several viewports coexist, only one has the current focus at a given moment. This viewport is highlighted to make it stand out.

User agents may render the same content in a variety of ways; each rendering is called a ***view***. For instance, a user agent may allow users to view an entire document or just a list of the document's headers. These are two different views of the document.

Visual-only presentation

A visual-only presentation is content consisting exclusively of one or more visual tracks presented concurrently or in series. A silent movie is an example of a visual-only presentation.

Visual track

A visual object is content rendered through a graphical viewport . Visual objects include graphics, text, and visual portions of movies and other animations. A visual track is a visual object that is intended as a whole or partial presentation. A visual track does not necessarily correspond to a single physical object or software object.

Voice browser

From "Introduction and Overview of W3C Speech Interface Framework" *[VOICEBROWSER]* : "A voice browser is a device (hardware and software) that interprets voice markup languages to generate voice output, interpret voice input, and possibly accept and produce other modalities of input and output."

Web resource

The term "Web resource" is used in this document in accordance with Web Characterization Terminology and Definitions Sheet *[WEBCHAR]* to mean anything that can be identified by a Uniform Resource Identifier (URI); refer to RFC 2396 *[RFC2396]* .

5. References

For the **latest version** of any W3C specification please consult the list of W3C Technical Reports at http://www.w3.org/TR/. Some documents listed below may have been superseded since the publication of this document.

Note: In this document, bracketed labels such as "[HTML4]" link to the corresponding entries in this section. These labels are also identified as references through markup.

5.1 How to refer to this document

There are two recommended ways to refer to the "User Agent Accessibility Guidelines 1.0" (and to W3C documents in general):

1. References to a specific version of "User Agent Accessibility Guidelines 1.0". For example, use the "this version" URI to refer to the current document: http://www.w3.org/TR/2002/WD-UAAG10-20020821/.
2. References to the latest version of "User Agent Accessibility Guidelines 1.0". Use the "latest version" URI to refer to the most recently published document in the series: http://www.w3.org/TR/UAAG10/.

In almost all cases, references (either by name or by link) should be to a specific version of the document. W3C will make every effort to make this document indefinitely available at its original address in its original form. The top of this document includes the relevant catalog metadata for specific references (including title, publication date, "this version" URI, editors' names, and copyright information).

An XHTML 1.0 *[XHTML10]* paragraph including a reference to this specific document might be written:

```
<p>
<cite><a href="http://www.w3.org/TR/2002/WD-UAAG10-20020821/">
"User Agent Accessibility Guidelines 1.0"</a></cite>,
I. Jacobs, J. Gunderson, E. Hansen, eds.,
W3C Working Draft, 21 August 2002.
The <a href="http://www.w3.org/TR/UAAG10/">latest
version</a> of this document is available at
http://www.w3.org/TR/UAAG10/.</p>
```

For very general references to this document (where stability of content, anchors, etc., is not required), it may be appropriate to refer to the latest version of this document. In this case, please use the "latest version" URI at the top of this document.

Other sections of this document explain how to build a conformance claim . Specification authors should also read the section on designing a including UAAG 1.0 requirements in other specifications .

5.2 Normative references

[CSS1]
: *"Cascading Style Sheets (CSS1) Level 1 Specification"*, B. Bos, H. Wium Lie, eds., 17 December 1996, revised 11 January 1999. This W3C Recommendation is http://www.w3.org/TR/1999/REC-CSS1-19990111.

[CSS2]
: *"Cascading Style Sheets, level 2 (CSS2) Specification"*, B. Bos, H. Wium Lie, C. Lilley, and I. Jacobs, eds., 12 May 1998. This W3C Recommendation is http://www.w3.org/TR/1998/REC-CSS2-19980512/.

[DOM2CORE]
: *"Document Object Model (DOM) Level 2 Core Specification"*, A. Le Hors, P. Le Hégaret, L. Wood, G. Nicol, J. Robie, M. Champion, S. Byrne, eds., 13 November 2000. This W3C Recommendation is http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/.

[DOM2STYLE]
: *"Document Object Model (DOM) Level 2 Style Specification"*, V. Apparao, P. Le Hégaret, C. Wilson, eds., 13 November 2000. This W3C Recommendation is http://www.w3.org/TR/2000/REC-DOM-Level-2-Style-20001113/.

[ECMASCRIPT]
: *"ECMAScript Language Specification"*, European Computer Manufacturers Association, December 1999. This specification is available at http://www.ecma.ch/ecma1/STAND/ECMA-262.HTM.

[INFOSET]
: *"XML Information Set"*, J. Cowan and R. Tobin, eds., 24 October 2001. This W3C Recommendation is http://www.w3.org/TR/2001/REC-xml-infoset-20011024/.

[JAVA]
: *"The Java Language Specification"*, Sun Microsystems Inc., J. Gosling, B. Joy, and G. Steele, September 1996. The specification is available at http://java.sun.com/docs/books/jls.

[RFC2046]
: *"Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types"*, N. Freed, N. Borenstein, November 1996.

[RFC2119]
: *"Key words for use in RFCs to Indicate Requirement Levels"*, S. Bradner, March 1997.

[WCAG10]
: *"Web Content Accessibility Guidelines 1.0"*, W. Chisholm, G. Vanderheiden, and I. Jacobs, eds., 5 May 1999. This W3C Recommendation is http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990505/.

5.3 Informative references

Some of the references in this section become normative if they are used to satisfy the requirements of guideline 6 and guideline 8.

[AT1998]
: The *Assistive Technology Act of 1998, 13 November 1998, United States P.L. 105-394*.

[ATAG10]
: *"Authoring Tool Accessibility Guidelines 1.0"*, J. Treviranus, C. McCathieNevile, I. Jacobs, and J. Richards, eds., 3 February 2000. This W3C Recommendation is http://www.w3.org/TR/2000/REC-ATAG10-20000203/.

[ATAG10-TECHS]
: *"Techniques for Authoring Tool Accessibility Guidelines 1.0"*, J. Treviranus, C. McCathieNevile, I. Jacobs, and J. Richards, eds., 4 May 2000. This W3C Note is http://www.w3.org/TR/2000/NOTE-ATAG10-TECHS-20000504/.

[CHARMOD]
: *"Character Model for the World Wide Web"*, M. Dürst and F. Yergeau, eds., 30 April 2002. This W3C Working Draft is http://www.w3.org/TR/2002/WD-charmod-20020430/. The latest version is available at http://www.w3.org/TR/charmod/.

[HTML4]
: *"HTML 4.01 Recommendation"*, D. Raggett, A. Le Hors, and I. Jacobs, eds., 24 December 1999. This W3C Recommendation is http://www.w3.org/TR/1999/REC-html401-19991224/.

[MATHML20]
: *"Mathematical Markup Language (MathML) Version 2.0"*, D. Carlisle, P. Ion, R. Miner, N. Poppelier, et al., 21 February 2001. This W3C Recommendation is http://www.w3.org/TR/2001/REC-MathML2-20010221/.

[PNG]
: *"PNG (Portable Network Graphics) Specification 1.0"*, T. Boutell, ed., 1 October 1996. This W3C Recommendation is http://www.w3.org/TR/REC-png.

[RDF10]
: *"Resource Description Framework (RDF) Model and Syntax Specification"*, O. Lassila, R. Swick, eds., 22 February 1999. This W3C Recommendation is http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[RFC2396]
: *"Uniform Resource Identifiers (URI): Generic Syntax"*, T. Berners-Lee, R. Fielding, L. Masinter, August 1998.

[RFC2616]
: *"Hypertext Transfer Protocol – HTTP/1.1"*, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, June 1999.

[SMIL]
: *"Synchronized Multimedia Integration Language (SMIL) 1.0 Specification"*, P. Hoschka, ed., 15 June 1998. This W3C Recommendation is http://www.w3.org/TR/1998/REC-smil-19980615/.

[SMIL20]
: *Synchronized Multimedia Integration Language (SMIL 2.0) Specification*, J. Ayars, et al., eds., 7 August 2001. This W3C Recommendation is http://www.w3.org/TR/2001/REC-smil20-20010807/.

[SVG]
: *"Scalable Vector Graphics (SVG) 1.0 Specification"*, J. Ferraiolo, ed., 2 August 2000. This W3C Candidate Recommendation is http://www.w3.org/TR/2000/CR-SVG-20000802/.

[UAAG10-CHECKLIST]
: An appendix to this document lists all of the checkpoints, sorted by priority. The checklist is available in either tabular form or list form.

[UAAG10-SUMMARY]
: An appendix to this document provides a summary of the goals and structure of User Agent Accessibility Guidelines 1.0.

[UAAG10-TECHS]
: *"Techniques for User Agent Accessibility Guidelines 1.0"*, I. Jacobs, J. Gunderson, E. Hansen, eds. The latest draft of the techniques document is available at http://www.w3.org/TR/UAAG10-TECHS/.

[UNICODE]
: *"The Unicode Standard, Version 3.2"*. This technical report of the Unicode Consortium is available at http://www.unicode.org/unicode/reports/tr28/. This is a revision of "The Unicode Standard, Version 3.0", The Unicode Consortium, Addison-Wesley Developers Press, 2000. ISBN 0-201-61633-5. Refer also to http://www.unicode.org/unicode/standard/versions/. For information about character encodings , refer to Unicode Technical Report #17 "Character Encoding Model".

[VOICEBROWSER]
: *"Introduction and Overview of W3C Speech Interface Framework"*, J. Larson, 4 December 2000. This W3C Working Draft is http://www.w3.org/TR/2000/WD-voice-intro-20001204/. The latest version is available at http://www.w3.org/TR/voice-intro/. This document includes references to additional W3C specifications about voice browser technology.

[W3CPROCESS]
: *"World Wide Web Consortium Process Document"*, I. Jacobs ed. The 19 July 2001 version of the Process Document is http://www.w3.org/Consortium/Process-20010719/. The latest version is available at http://www.w3.org/Consortium/Process/.

[WCAG10-TECHS]
: *"Techniques for Web Content Accessibility Guidelines 1.0"*, W. Chisholm, G. Vanderheiden, and I. Jacobs, eds., 6 November 2000. This W3C Note is http://www.w3.org/TR/1999/WAI-WEBCONTENT-TECHS-19990505/. The latest version is available at http://www.w3.org/TR/WCAG10-TECHS/. Additional format-specific techniques documents are available from this Note.

[WEBCHAR]
: *"Web Characterization Terminology and Definitions Sheet"*, B. Lavoie, H. F. Nielsen, eds., 24 May 1999. This is a W3C Working Draft that defines some

terms to establish a common understanding about key Web concepts. This W3C Working Draft is http://www.w3.org/1999/05/WCA-terms/01.

[XAG10]
: *"XML Accessibility Guidelines 1.0"*, D. Dardailler, S. Palmer, eds., 29 August 2001. This W3C Working Draft is http://www.w3.org/TR/2001/WD-xmlgl-20010829. The latest version is available at http://www.w3.org/TR/xmlgl.

[XHTML10]
: *"XHTML[tm] 1.0: The Extensible HyperText Markup Language"*, S. Pemberton, et al., 26 January 2000. This W3C Recommendation is http://www.w3.org/TR/2000/REC-xhtml1-20000126/.

[XML]
: *"Extensible Markup Language (XML) 1.0"*, T. Bray, J. Paoli, C.M. Sperberg-McQueen, eds., 10 February 1998. This W3C Recommendation is http://www.w3.org/TR/1998/REC-xml-19980210.

[XMLDSIG]
: *"XML-Signature Syntax and Processing"*, D. Eastlake, J. Reagle, D. Solo, eds., 12 February 2002. This W3C Recommendation is http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/.

[XMLENC]
: *"XML Encryption Syntax and Processing"*, D. Eastlake, J. Reagle, eds., 4 March 2002. This W3C Candidate Recommendation is http://www.w3.org/TR/2002/CR-xmlenc-core-20020304/. The latest version is available at http://www.w3.org/TR/xmlenc-core/.

6. Acknowledgments

The active participants of the User Agent Accessibility Guidelines Working Group who authored this document were: James Allan (Texas School for the Blind and Visually Impaired), Denis Anson (College Misericordia), Harvey Bingham, Jon Gunderson (Chair of the Working Group, University of Illinois, Urbana-Champaign), Eric Hansen (Educational Testing Service), Ian Jacobs (Team Contact, W3C), Tim Lacy (Microsoft), David Poehlman, and Rich Schwerdtfeger (IBM).

Many thanks to the following people who have contributed through review and past participation in the Working Group: Paul Adelson, Jonny Axelsson, Kitch Barnicle, Olivier Borius, Judy Brewer, Dick Brown, Bryan Campbell, Kevin Carey, Tantek Çelik, 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, Glen Gordon, John Grotting, Markku Hakkinen, Earle Harrison, Chris Hasser, Kathy Hewitt, Philipp Hoschka, Masayasu Ishikawa, Phill Jenkins, Earl Johnson, Jan Kärrman (for help with html2ps), Leonard Kasday, George Kerscher, Marja-Riitta Koivunen, Peter Korn, Josh Krieger, Catherine Laws, Aaron Leventhal, Greg Lowney, Susan Lesch, Scott Luebking, William Loughborough, Napoleon Maou, Matt May, Charles McCathieNevile (W3C), Peter Meijer, Karen Moses, Dirk Mueller, Masafumi Nakane, Mark Novak, Charles Oppermann, Mike Paciello, David Pawson, Michael Pederson, Helen Petrie, Michael Pieper, Richard Premack, Mickey Quenzer, Jan Richards, Hans Riesebos, Joe Roeder, Lakespur L. Roca, Gregory Rosmaita, Madeleine Rothberg, Lloyd Rutledge, Liam Quinn, T.V. Raman, Robert Savellis, 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.