W3C WD-WAI-AUTOOLS-19990128

WAI Authoring Tool Guidelines

W3C Working Draft *28-JAN-1999*

This version:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128
Latest version:
http://www.w3.org/TR/WD-WAI-AUTOOLS
Previous version:
http://www.w3.org/TR/1998/WD-WAI-AUTOOLS-19981112

Editors:
Jutta Treviranus <jutta.treviranus@utoronto.ca>
Jan Richards <jan.richards@utoronto.ca>
Ian Jacobs <ij@w3.org>
Charles McCathieNevile <charles@w3.org>

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

Abstract

This document provides guidelines to authoring tool manufacturers or developers. The purpose of this document is two-fold: to assist developers in designing authoring tools that generate accessible Web content and to assist developers in creating an accessible authoring tool user interface. Accessible Web content is achieved by encouraging authoring tool users to create accessible Web content (through mechanisms such as prompts, alerts, checking and repair functions, help files and automated tools), but also by ensuring that the automatic processes of the authoring tool generate accessible content. This will result in authoring tools that can be used by a broader range of users and in the proliferation of Web pages that can be read by a broader range of readers.

This document is part of a series of accessibility documents published by the W3C Web Accessibility Initiative.

Status of this document

This is a W3C Working Draft of the W3C WAI Authoring Tool Guidelines. for review by W3C Members and other interested parties. It is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". This is work in progress and does not imply endorsement by, or the consensus of, either W3C or members of the WAI Authoring Tool (AU) Working Group.

Editors' note. *There are a number of (normative) checkpoints in this document. Some of these are expected to become (informative) techniques in subsequent drafts.*

The goals of the WAI AU Working Group are discussed in the WAI AU charter. A list of the current AU Working Group members is available.

Available formats

This document is available in the following formats:

HTML:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.html
A plain text file:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.txt,
HTML as a gzip'ed tar file:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.tgz,
HTML as a zip file (this is a '.zip' file not an '.exe'):
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.zip,
A PostScript file:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.ps,
A PDF file:
http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.pdf.

In case of a discrepancy between the various formats of the specification, http://www.w3.org/TR/1999/WD-WAI-AUTOOLS-19990128/wai-autools.html is considered the definitive version.

Comments

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

Table of Contents

1 Introduction 4
1.1 Checkpoint Priorities 4
2 Ensure that content produced by the tool is accessible 4
2.1 Generate standard markup 4
2.2 Support all accessible content recommendations 4
2.3 Identify and allow the user to correct all inaccessible markup . . .5
2.4 Ensure that all markup inserted by the authoring tool is accessible . .5
2.5 Provide mechanism for managing alternative content 6
2.6 Never remove existing accessible structure 7
3 Encourage Authors to Create Accessible Documents 7
3.1 Emphasize accessible authoring practices 7
3.2 Provide comprehensive accessibility help to authors 7
3.3 Provide rationales which stress Universal Design 8
3.4 Promote accessibility in all Help examples 8
3.5 Ensure that users may configure accessibility mechanisms 9
3.6 Integrate accessibility solutions naturally 9
3.7 Provide the author with progress feedback 9
4 Ensure the Authoring Tool is Accessible to Users with Disabilities . . 10
4.1 Provide optional views of the edited document 10
4.2 Provide text representations of elements 10
4.3 Offer text representations of site maps 10
5 Sample Implementations 11
5.1 Alt-Text for the HTML 4.0 IMG Element 11
5.2 Tool: "Alt"-text registry 12
6 Terms and Definitions 12
6.1 Integrated Author Guidance and Prompting 12
6.2 Alert Techniques 13
6.3 Markup Editing Tools and Functions 13
6.4 Documents, Elements, and Attributes 14
6.5 Accessibility Terms 15
6.6 Alternative Representation of Content 15
6.7 Inserting and Editing 15
6.8 Alert Techniques 16
6.9 Selection, Focus, and Events 16
7 Acknowledgments 16
8 References 16

1 Introduction

The guidelines in this document are meant to help authoring tool developers and vendors design products that encourage authors to adopt accessible authoring practices. For the purposes of this document the term "authoring tool" will refer to authoring tools [p. 13] , generation tools [p. 13] and conversion tools [p. 13] . These guidelines emphasize the role of the user interface in informing, supporting, correcting and motivating authors during the editing process. For a more detailed discussion of accessible Web authoring practices, see the WAI Page Author Guidelines [p. 17] document.

1.1 Checkpoint Priorities

Each checkpoint in this document is assigned a priority.

[Priority 1]
: This checkpoint is fundamental to the accessibility of authoring tools, or to the creation of accessible documents by authoring tools.

[Priority 2]
: This checkpoint is important to the accessibility of authoring tools, or to the creation of accessible documents by authoring tools.

[Priority 3]
: This checkpoint promotes the accessibility of authoring tools, or the creation of accessible documents by authoring tools.

This document also refers to guidelines, checkpoints, and techniques defined in the WAI Page Author Guidelines [p. 17] and to priorities assigned to them (indicated, for example, by [Page-Author-Priority 1] [p. 5]).

2 Ensure that content produced by the tool is accessible

Authoring Tools are used to automate the low-level tasks involved in producing Web pages. The power of this automation can enhance the accessibility of the Web if is used to ensure that the code produced promotes accessibility, and frees the author to concentrate on the higher level problems of overall design, content, description, etc.

2.1 Generate standard markup

The first step towards accessibility is interoperability.

2.1.1: [Priority 1]
: Ensure that content is created in accordance with W3C specifications (or other standards when applicable).

2.2 Support all accessible content recommendations

Methods for ensuring accessible markup vary with different markup languages.

2.2.1: [Priority 1]
: Support all accessibility features that have been defined for the markup language(s) supported by the tool.

Listing the accessibility features of specific languages lies beyond the scope of this document. However, an informative list of documents that address accessible Web authoring practices follows.

Page Author Accessibility Features: (The actual accessible markup solutions)

- General: WAI Page Authoring Guidelines: Techniques [p. 17]
- HTML4: HTML4 Accessibility Improvements [p. 17]
- CSS2: CSS2 Accessibility Improvements [p. 17]
- SMIL, MathML

Page Author Implementation Priorities: (The priorities placed on the accessibility markup solutions)

- General: WAI Page Author Guidelines [p. 17]

Mechanisms that can be employed by authoring tools to support accessible authoring practices are discussed in Section 5 [p. 11] .

2.3 Identify and allow the user to correct all inaccessible markup

Many authoring tools allow their users to create documents with little or no knowledge about the underlying markup. To ensure accessibility, authoring tools must be designed so that they may automatically identify inaccessible content, even when the markup itself is hidden from the author.

Checkpoints:

2.3.1: [Priority 1]
Check existing documents when they are opened for editing.

2.3.2: [Priority 1]
Check documents during all types of editing [p. 15] (including hand-coding, paste operations, and code insertions).

2.3.3: [Priority 1]
Alert the author (according to a user-configurable schedule) when problems are detected. See the sections on ensuring that users may configure accessibility mechanisms [p. 9] and Alert Techniques. [p. 13]

2.3.4: [Priority 1]
Assist authors in correcting accessibility problems without requiring them to know the details of the markup language or its accessibility features.

2.4 Ensure that all markup inserted by the authoring tool is accessible

If markup is automatically generated, the author will be unaware of the accessibility status of the final product unless they expend extra effort to make appropriate corrections by hand. Since most authors are unfamiliar with accessibility, these problems are likely to remain.

Checkpoints:

2.4.1: [Priority 1]
Automated markup insertion functions [p. 14] must make use of appropriate accessible solutions, even if this means presenting the author with extra prompts [p. 16] for necessary information or structure during or following the process.
2.4.2: [Priority 1]
Do not produce inaccessible markup [p. 15] .

2.5 Provide mechanism for managing alternative content

Textual descriptions, including "alt"-text, long descriptions, video captions, and transcripts are absolutely necessary for the accessibility of all images, applets, video, and audio files. However, the task of writing these descriptions is probably the most time-consuming accessibility recommendation made to the author.

Including professionally written descriptions for all multimedia files (e.g., clip art) packed with the software:

1. will save users time and effort
2. will cause a significant number of professionally written descriptions to circulation the Web
3. will provide users with convenient models to emulate when they write their own descriptions
4. will show authors the importance of description writing

Checkpoints:

2.5.1: [Priority 2]
Include professionally written descriptions for all multimedia files packaged with the authoring tool.
2.5.2: [Priority 2]
Suggest pre-written descriptions as default text whenever one of the associated files is inserted into the author’s document.
2.5.3: [Priority 2]
Allow authors to make keyword searches of the description database (to simplify the task of finding relevant images).
2.5.4: [Priority 1]
Prompt the user, on a configurable schedule, to provide "alt"-text for images, image maps, and image map links.
2.5.5: [Priority 1]
Prompt the author to provide a caption or transcription for any audio segment.
2.5.6: [Priority 1]
Prompt the author to provide a caption or transcription for any video segment.
2.5.7: [Priority 1]
Allow the author to provide a long description for any graphic element.
2.5.8: [Priority 1]
Do not generate description text or insert place-holder text.
2.5.9: [Priority 1]
Insert (automatically) pertinent human-authored description text when the meaning or function of the described object is known with certainty.

2.6 Never remove existing accessible structure

Many applications feature the ability to convert documents from other formats (e.g., Rich Text Format) into a markup format, such as HTML. Markup changes may also be made to facilitate efficient editing and manipulation. These processes are usually hidden from the user's view and may create inaccessible content or cause inaccessible content to be produced.

Checkpoints:

2.6.1: [Priority 1]
Generate documents that respect the WAI Page Author Guidelines [p. 17] .

2.6.2: [Priority 1]
Never remove or modify structure or content that is necessary for continued accessibility.

2.6.3: [Priority 1]
Provide a summary of all automated structural changes that may affect accessibility.

3 Encourage Authors to Create Accessible Documents

Help files, accompanying documentation, and the design of the user interface can all influence the way an author uses a tool. Appropriate materials can educate authors who are unsure of what accessibility is, and demonstrate ways to improve it. Including accessibility-related features in examples, and explaining how to use those features, and why they are important, can all help promote the goal of accessible design to an author.

3.1 Emphasize accessible authoring practices

Recommended accessible authoring practices [p. 5] (and their priorities [p. 5]) must be taken into account during the design of relevant user interface components and program functionality.

Checkpoints:

3.1.1: [Priority 1]
Do not encourage or recommend those authoring practices discouraged by [Page-Author-Priority 1].

3.1.2: [Priority 1]
Ensure that the highest-priority authoring practices are the most visible and easily initiated by the author.

3.2 Provide comprehensive accessibility help to authors

The issues surrounding Web accessibility are often unknown to Web authors. Providing convenient links to clear and concisely written help files will contribute to author acceptance of, and education about, markup accessibility. The accessibility help files should explain the accessibility problem or accessibility feature quickly,

Checkpoints:

3.2.1: [Priority 1]
Implement context-sensitive help for all special accessibility terms, as well as tasks related to accessibility.

3.2.2: [Priority 1]
Link those mechanisms used to identify accessibility problems (e.g., icons, outlining or other emphasis within the user interface) to help files.
3.2.3: [Priority 1]
In help text, emphasize available solutions rather than on elements that have been incorrectly marked up.
3.2.4: [Priority 1]
In help text, provide numerous examples.
3.2.5: [Priority 1]
Link from help text to any automated correction utilities.

3.3 Provide rationales which stress Universal Design

Most users are unfamiliar with accessibility issues on the Web. By incorporating explanations of universal design benefits into authoring tools, authors will better understand the value of accessible page design. The Universal Design principles of supporting flexible display and control choices, are critical for:

- hands-free, eyes-free, voice-activated browsing devices such as Web phones
- the large number of slow Web connections
- Web users who prefer text-only browsing to avoid "image clutter"
- the aging population (with the accompanying decrease in visual, hearing, motor, and cognitive abilities)
- the relatively high Web presence of people with sensory and motor disabilities.

Checkpoints:

3.3.1: [Priority 1]
In help text, explain the importance of utilizing accessibility features generally and for specific instances.
3.3.2: [Priority 1]
In help text, emphasize accessibility features that benefit multiple groups.

For more information on Universal Design, visit the Trace Center.

3.4 Promote accessibility in all Help examples

In addition to a help section dedicated to accessibility [p. 7] , accessibility principles should be followed for *all* applicable markup examples in the rest of the help system. This will increase integration and help show authors that accessibility is a normal part of authoring, rather than a separate concern.
Checkpoints:

3.4.1: [Priority 3]
Ensure that accessibility solutions accompany descriptions of accessible markup practices (ex. IMG elements should appear with "alt"-text).
3.4.2: [Priority 3]
In help text, provide even low-priority accessibility solutions.

3.5 Ensure that users may configure accessibility mechanisms

In supporting the creation of accessible Web content, authoring tools must take into account the differing authoring styles of their users. Some users may prefer to be alerted to problems when they occur, whereas others may prefer to perform a check after the document is completed. This is analogous to programming environments that allow users to decide whether to check for correct code during editing or at compile time.

Checkpoints:

3.5.1: [Priority 1]
Allow users to control both the nature and timing of accessibility alerts (for a given set of options).

3.5.2: [Priority 1]
Allow users to choose different alert levels based on the priority of authoring accessibility recommendations. (Specifically, the user should have the option of determining the extent of alerts for [Page-Author-Priority 2] [p. 5] and [Page-Author-Priority 3] [p. 5] recommendation items.)

3.5.3: [Priority 1]
Do not allow users to disable non-intrusive alerts for [Page-Author-Priority 1] [p. 5] items.

3.6 Integrate accessibility solutions naturally

When a new feature is added to an existing software tool without proper integration, the result is often an obvious discontinuity. Differing color schemes, fonts, interaction styles and even application stability can be factors affecting user acceptance of the new feature.

Checkpoints:

3.6.1: [Priority 2]
Integrate accessibility features into the overall "look and feel" of the authoring tool.

3.6.2: [Priority 2]
Ensure that accessibility features never interfere with any of the expected operations of an author’s editing environment. For example, *fundamental* operations such as saving, closing, and pasting should not be canceled or postponed due to the existence of accessibility problems.

3.7 Provide the author with progress feedback

Achieving accessibility requires some extra effort and cooperation from the author. In order to maintain user goodwill and acceptance of accessible authoring practices, the user should receive progress feedback regarding satisfied accessibility objectives.

Checkpoints:

3.7.1: [Priority 1]
Provide the user with progress feedback as accessibility goals are accomplished.

4 Ensure the Authoring Tool is Accessible to Users with Disabilities

Principles to consider in making the authoring tool accessible to authors with disabilities relate to 3 classes of functionality:

1. The authoring tool is a software program with standard user interface elements and as such should follow relevant user interface accessibility guidelines (links to TRACE guidelines, Microsoft, SUN, DACX, Apple, IBM guidelines)
2. The authoring tool frequently encompasses the functionality of a user agent or browser and as such should follow the User Agent Guidelines [p. 17] .
3. The authoring tool has unique functionality as a Web content editor. Access to this unique functionality will be addressed in these guidelines.

4.1 Provide optional views of the edited document

When creating or editing a Web page the desired ultimate rendering of the page may not be optimal for creating and editing.

Checkpoints:

4.1.1: [Priority 1]

Support at least two views:

1. an authoring/editing view
2. a publishing or browser view, (similar to the normal and page view or print preview of popular word processors).

4.1.2: [Priority 1]

Ensure that the styles used to author are independent of those used for the published document (e.g., the font size, letter and line spacing, and text and background color, etc.).

4.2 Provide text representations of elements

Graphically represented elements cannot be identified by third-party assistive technologies that translate text to Braille, speech, or large print. Some authoring tools display the start and end tags as graphics.

Checkpoints:

4.2.1: [Priority 1]

Allow the author to display start and end tags in a text format.

4.2.2: [Priority 1]

Surround start and end tags with text brackets to help distinguish them from the remainder of the document.

4.3 Offer text representations of site maps

Graphic representation of Web pages or Web site elements in site management tools cannot be identified by third-party assistive technologies that translate text to Braille, speech, or large print.

Checkpoints:

4.3.1: [Priority 1]
Allow the author to display the site map in text form (e.g., as a structured tree file).

5 Sample Implementations

The Sample Implementations are *not* guidelines. The section has been included to illustrate how the design principles embodied in the guidelines sections can be applied to concrete issues. The specific ideas discussed in this section are meant to be used only as clarification.

5.1 Alt-Text for the HTML 4.0 IMG Element

"Alt"-text is generally considered the most important aid to accessibility. For this reason, the issue of "alt"-text has been chosen as the subject for the first sample implementation.

2.1 Generate standard markup [p. 4]
Implementation: In any content produced, the IMG element is always properly formed as defined in the HTML4 specification. This means that the element contains both a "src" attribute and an "alt" attribute.
2.2 Support all accessible content recommendations [p. 4]
Implementation: Due the [Page-Author-Priority 1] [p. 5] recommendation status of "alt"-text in the WAI Page Author Guidelines, special attention will be devoted to prompting and guiding the user toward full "alt" coverage.
3.1 Emphasize accessible authoring practices [p. 7]
Implementation: The "alt" attribute appears immediately below the "src" attribute in the image properties listing. Whenever the properties for an image without "alt"-text are examined, visual highlighting of the "alt" entry field remind the user that "alt"-text should not be left empty. In addition, when an image without "alt"-text is selected, *Insert "alt"-text* is one of the options presented to the user.
2.3 Identify and allow the user to correct all inaccessible markup [p. 5]
Implementation: If the user opens content or pastes in markup containing an IMG element that lacks "alt"-text, the author is prompted to add them (unless they have configured the tool to postpone this task).
3.2 Provide comprehensive accessibility help to authors [p. 7]
Implementation: Whenever missing "alt"-text is flagged (anywhere in the tool suite) the same quick explanation, extended help, and examples are offered.
2.5 Provide mechanism for managing alternative content [p. 6]
Implementation: The authoring tool is shipped with many ready-to-use clip art and other images. For each of these images a short "alt"-text string and a longer description have been pre-written and stored in the "alt"-text registry [p. 12] .
2.4 Ensure that all markup inserted by the authoring tool is accessible [p. 5]
Implementation: If the user drags an image from the desktop into the authoring tool, the user will be prompted for "alt"-text for the IMG element (unless the user has postponed this task).
2.6 Never remove existing accessible structure [p. 7]
Implementation: The authoring tool has the capability of opening and converting word processor documents into HTML. If an image is encountered during this process, the user will be prompted for "alt"-text. The authoring tool sometimes makes changes to the HTML it works with to allow more efficient manipulation.

These changes *never* result in the removal or modification of "alt"-text entries.

3.5 Ensure that users may configure accessibility mechanisms [p. 9]
Implementation: A configuration system allows the user to decide whether they wish to be reminded each time they place an IMG element without "alt"-text or if they will complete the "alt"-text entry task at a later time. The configuration system does not contain the option of disabling "alt"-text checking completely. Other options allow the user to specify the behavior of the "alt"-text registry [p. 12] .

3.6 Integrate accessibility solutions naturally [p. 9]
Implementation: At no point do "alt"-text requests appear *on their own* or in a non-standard manner. Instead "alt"-text notices and emphasis appear as integrated and necessary as the "src" attribute.

3.3 Provide rationales which stress Universal Design [p. 8]
Implementation: In addition to describing the need for "alt"-text for access by people with visual disabilities, the rationales mention how "alt"-text allows users of Web phones and other non-visual browsing technologies to access the content of the image.

3.4 Promote accessibility in all Help examples [p. 8]
Implementation: Whenever the IMG element appears in the help system, the "alt" attribute is always present. Links to "alt"-text specific help and rationale are provided.

3.7 Provide the author with progress feedback [p. 9]
Implementation: Whenever an accessibility checker completes a run, a summary list of accessibility issues is presented. When the user has entered "alt"-text for all the images in a document, the "alt"-text completed box will be checked in the summary. This box will remain checked as long as no images without "alt"-text are added.

5.2 Tool: "Alt"-text registry

This tool does not have a visual window presence as far as the user is concerned. It works by saving an association between every "alt"-text label that a user writes with the name of the image, applet, image map, or image map link. Then, whenever one of these elements is inserted, the file name information of the object is checked against the registry association file. If a match is found, then the pre-written "alt" text is displayed as a default choice, allowing users to avoid the repetition of writing multiple descriptions for the same image. The ability to store several descriptions in different languages might also be supported. In more sophisticated implementations, the tool may include a prediction algorithm that takes into account the recency of the "alt"-text, name similarity, and target similarity when searching for matches. This tool has the curb-cut advantage that the descriptions (especially the professionally written ones that come with bundled images) will allow users to search images using keyword searches, thereby simplifying the task of finding appropriate images.

6 Terms and Definitions

6.1 Integrated Author Guidance and Prompting

Interface mechanisms such as dialogs, menus, toolbars, and palettes can be structured so that markup or elements that are accessible are given as the first and easiest choice.

Prompts can be used to encourage authors to provide information needed to make the content accessible (such as alternative textual representations). Prompts are simple requests for information before a markup structure has been finalized. For example, an "alt"-text entry field prominently displayed in an image insertion dialog would constitute a prompt. Prompts are relatively unintrusive and address a problem before it has been committed. However, once the user has ignored the prompt, its message is unavailable.

6.2 Alert Techniques

Alerts warn the author that there are problems that need to be addressed. The art of attracting users' attention is a tricky issue. The way in which users are alerted, prompted, or warned will influence their view of the tool as well as their opinion of accessible authoring.

The following are sample alert possibilities with a short definition and a brief discussion of their advantages and disadvantages.

Interruptive Alerts
: Interruptive alerts are informative messages that interrupt the edit process for the user. For example, interruptive alerts are often presented when a user's action could cause a loss of data. Interruptive alerts allow problems to be brought to the user's attention immediately. However, users may resent the constant delays and forced actions. Many people prefer to finish expressing an idea before returning to edit its format.

Unintrusive Alerts
: Unintrusive alerts are alerts such as icons, underlines, and gentle sounds that can be presented to the user without necessitating immediate action. for example, in some word processors misspelled text is highlighted without forcing the user to make immediate corrections. These alerts allow users to continue editing with the knowledge that problems will be easy to identify at a later time. However, users may become annoyed at the extra formatting or may choose to ignore the alerts altogether.

6.3 Markup Editing Tools and Functions

Authoring Tool
: An *Authoring Tool* is any application that is specifically designed to aid users in editing markup and presentation language documents. The editing processes covered by this definition may range from direct hand coding (with automated syntax support or other markup specific features) to WYSIWYG editors that do not present the actual underlying markup to the author for editing. This definition does *not* include text editors and word processors that also allow HTML to be hand produced.

Conversion Tool
: A *Conversion Tool* is any application or application feature that allows content in some other format (proprietary or not) to be converted automatically into a particular markup language. This includes software whose primary function is to convert documents to a particular markup language as well as "save as HTML" (or other markup language) features in non-markup applications.

Generation Tool
: A *Generation Tool* is a program or script that produces automatic markup "on the fly" by following a template or set of rules. The generation may be performed on either the server or client side.

Site Management Tool
A tool that provides an overview of an entire Web site indicating hierarchical structure. It will facilitate management through functions that may include automatic index creation, automatic link updating, and broken link checking.

Publishing Tool
A tool that allows content to be uploaded in an integrated fashion. Sometimes these tools makes changes such as local hyper-reference modifications. Although these tools sometimes stand alone, they may also be integrated into site management tools.

Image Editor
A graphics program that provides a variety of options for altering images of different formats.

Video Editor
A tool that facilitates the process of manipulating video images. Video editing includes cutting segments (trimming), re-sequencing clips, and adding transitions and other special effects.

Multi-media Authoring Tool
Software that facilitates integration of diverse media elements into an comprehensive presentation format. May incorporate video, audio, images, animations, simulations, and other interactive components.

Automated Markup Insertion Function
Automated markup insertion functions are the features of an authoring tool that allow the user to produce markup without directly typing it. This includes a wide range of tools from simple markup insertion aids (such as a bold button on a toolbar) to markup managers (such as table makers that include powerful tools such as "split cells" that can make multiple changes) to high level site building wizards that produce almost complete documents on the basis of a series of user preferences.

6.4 Documents, Elements, and Attributes

Document
A *document* is a series of elements that are defined by a language (e.g., HTML 4.0 or an XML application).

Element
Each *element* consists of a name that identifies the type of element, optional attributes that take values, and (possibly empty) content.

Attributes
Some *attributes* are integral to document accessibility (e.g., the "alt", "title", and "longdesc" attributes in HTML).

Rendered Content
The *rendered content* is that which an element actually causes to be rendered by the user agent. This may differ from the element’s structural content. For example, some elements cause external data to be rendered (e.g., the IMG element in HTML), and in some cases, browsers may render the value of an attribute (e.g., "alt", "title") in place of the element’s content.

6.5 Accessibility Terms

Accessibility Awareness

The term accessibility awareness is used to describe an application that has been designed to maximize the ease of use of the interface and its products for people with differing needs, abilities and technologies. In the case of authoring tools, this means that (1) care has been taken to ensure that the content produced by user-authors is accessible and (2) that the user interface has been designed to be usable with a variety of display and control technologies.

Inaccessible Markup, Inaccessible Element, Inaccessible Attribute, Inaccessible Authoring Practice and Access Barrier

All these terms are used in the context of inaccessibility as defined by the WAI Page Author Guidelines [p. 17] .

Accessibility Solution, Accessible Authoring Practice

These terms refer to markup techniques than can be used to eliminate or reduce accessibility problems as they are defined above.

6.6 Alternative Representation of Content

Alternate Textual Representations

Certain types of content may not be accessible to all users (e.g., images), so authoring tools must ensure that *alternate textual representations* ("Alt-text") of information is available to the user. Alternate text can come from element content (e.g., the OBJECT element) or attributes (e.g., "alt" or "title").

Description Link (D-link)

A description link, or *D-Link*, is an author-supplied link to additional information about a piece of content that might otherwise be difficult to access (image, applet, video, etc.).

Transcripts

A transcript is a line by line record of all dialog and action within a video or audio clip.

Video Captions

A video caption is a textual message that is stored in the text track of a video file. The video caption describes the action and dialog for the scene in which it is displayed.

6.7 Inserting and Editing

Inserting an element

Inserting an element involves placing that element's markup within the markup of the file. This applies to all insertions, including, but not limited to, direct coding in a text editing mode, choosing an automated insertion from a pull-down menu or tool bar button, "drag-and-drop" style insertions, or "paste" operations.

Editing an element

Editing an element involves making changes to one or more of an element's attributes or properties. This applies to all editing, including, but not limited to, direct coding in a text editing mode, making changes to a property dialog or direct UI manipulation.

6.8 Alert Techniques

Prompts
Prompts are simple requests for information before a markup structure has been finalized.

Interruptive Alerts
Interruptive alerts are informative messages that interrupt the edit process for the user.

Unintrusive Alerts
Unintrusive alerts are alerts such as icons, underlines, and gentle sounds that can be presented to the user without necessitating immediate action.

Alert Tools
Alert tools allow a batch detection process to address all problems at a given time.

6.9 Selection, Focus, and Events

Views
An authoring tool may offer several *views* of the same document. For instance, one view may show raw markup, a second may show a structured tree view, a third may show markup with rendered objects while a final view shows an example of how the document may appear if it were to be rendered by a particular browser.

Selection
A *selection* is a set of elements identified for a particular operation. The user selection identifies a set of elements for certain types of user interaction (e.g., cut, copy, and paste operations). The user selection may be established by the user (e.g., by a pointing device or the keyboard) or via an accessibility API. A view may have several selections, but only one user selection.

Current User Selection
When several views co-exist, each may have a user selection, but only one is active, called the *current user selection*. The selections may be rendered specially (e.g., visually highlighted).

Focus
The *focus* designates the active element (e.g., link, form control, element with associated scripts, etc.) in a view that will react when the user next interacts with the document.

7 Acknowledgments

Many thanks to the following people who have contributed through review and comment: Harvey Bingham, Judy Brewer, Carl Brown, Wendy Chisholm, Daniel Dardailler, Phill Jenkins, William Loughborough, Charles Oppermann, and Gregg Vanderheiden.

If you have contributed to the AU guidelines and your name does not appear please contact the editors to add your name to the list.

8 References

[HTML40]
"HTML 4.0 Recommendation", D. Raggett, A. Le Hors, and I. Jacobs, eds. The HTML 4.0 Recommendation is available at:

http://www.w3.org/TR/REC-html40/

[CSS1]

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

[CSS2]

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

[WAI-PAGEAUTH]

"WAI Page Authoring Guidelines", G. Vanderheiden, W. Chisholm, and I. Jacobs, eds. These guidelines for designing accessible documents are available at:
http://www.w3.org/TR/WD-WAI-PAGEAUTH/

[Page Authoring Techniques]

"Techniques for WAI Page Authoring Guidelines", G. Vanderheiden, W. Chisholm, and I. Jacobs, eds. These guidelines for designing accessible documents are available at:
http://www.w3.org/TR/WD-WAI-PAGEAUTH/wai-pageauth-tech

[WAI-USERAGENT]

"WAI User Agent Guidelines", J. Gunderson and I. Jacobs, eds. These guidelines for designing accessible user agents are available at:
http://www.w3.org/TR/WD-WAI-USERAGENT/

[CSS2-ACCESS]

"WAI Resources: CSS2 Accessibility Improvements", I. Jacobs and J. Brewer, eds. This document, which describes accessibility features in CSS2, is available at:
http://www.w3.org/WAI/References/CSS2-access

[HTML4-ACCESS]

"WAI Resources: HTML 4.0 Accessibility Improvements", I. Jacobs, J. Brewer, and D. Dardailler, eds. This document, which describes accessibility features in HTML 4.0, is available at:
http://www.w3.org/WAI/References/HTML4-access

[Access Aware Authoring Tools]

"The Three-tions of Accessibility-Aware HTML Authoring Tools", J. Richards. Available at:
http://www.utoronto.ca/atrc/rd/hm/3tions.htm