

# W3C Techniques for Authoring Tool Accessibility

W3C

This document is published for review by the Authoring Tool Accessibility Guidelines Working Group (AUWG) and other interested parties, and has not been endorsed by the Working Group, the W3C or any of the W3C Membership.

For further information about Working Group decisions, please consult the minutes of AUWG Meetings.

This document has been produced by the Authoring Tool Accessibility Guidelines Working Group (AUWG) as part of the Web Accessibility Initiative (WAI). The goals of the Working Group are discussed in the AUWG charter.

Please send general comments about this document to the public mailing list: [w3c-wai-au@w3.org](mailto:w3c-wai-au@w3.org) (public archives).

A list of current W3C Recommendations and other technical documents including Working Drafts and Notes can be found at <http://www.w3.org/TR>.

# Table of Contents

Abstract	. . . . .	.
Status of this document	. . . . .	.
1 Introduction	. . . . .	.4
1.1 How the Techniques are organized	. . . . .	.4
2 Guidelines	. . . . .	.5
1. Support accessible authoring practices.	. . . . .	.5
2. Generate standard markup.	. . . . .	.8
	. . . . .	.9

# 1 Introduction

The "Authoring Tool Accessibility Guidelines 1.0" [ATAG10] has two goals: to assist developers in designing authoring tools that produce accessible Web content and to assist developers in creating an accessible authoring interface. The present "Techniques Document" suggests to developers some strategies for meeting those goals.

Implementation of techniques for some of these guidelines requires familiarity with the Web Content Accessibility Guidelines (WCAG) 1.0 [WCAG10] . In addition, readers are strongly encouraged to become familiar with the "Techniques for Web Content Accessibility Guidelines 1.0" [WCAG10-TECHS] and "Techniques for User Agent Accessibility Guidelines 1.0" [UAAG10-TECHS] .

**Note:** The techniques in this document are merely suggestions; they are not required for conformance to "Authoring Tool Accessibility Guidelines 1.0". These techniques are not necessarily the only way of satisfying the checkpoint, nor are they necessarily a definitive set of requirements for satisfying a checkpoint.

## 1.1 How the Techniques are organized

This document has the same structure as the "Authoring Tool Accessibility Guidelines 1.0" [ATAG10] : seven guidelines, each of which includes at least one checkpoint. Information about checkpoint priorities is found in the "Authoring Tool Accessibility Guidelines 1.0".

Unlike "Authoring Tool Accessibility Guidelines 1.0", the current document includes a list of techniques after each checkpoint. Techniques may be suggested strategies, references to other accessibility resources (noted "Reference"), or examples of how deployed tools satisfy the checkpoint (noted "Sample").

For some guidelines there are techniques or information that are relevant to the entire guideline. These are provided at the end of the section for the relevant guideline.

Some of the sample techniques describe how Amaya satisfies the checkpoints. Amaya [AMAYA]

## 2 Guidelines

Guideline 1. Support accessible authoring practices.



markup for known structures. For example, use list markup of an appropriate type rather than creating multiple line paragraphs and beginning each line with an image of a bullet. Do not use list markup for an indentation effect.

- Do not publish Web content in markup languages that do not allow for equivalent alternative information to be included for media-specific

Guideline 2. Generate standard markup.





format could also be published as an HTML document. Tools that dynamically generate Web content may use HTTP content negotiation to facilitate this.

- Satisfying checkpoint 3.5 would provide much of the required functionality. Refer also to checkpoint 4.1. Refer also to checkpoint 6.2..
- **Sample:** Amaya prompts the author to provide equivalent text for `IMG` and `AREA` elements, and
- Some Techniques listed for different languages, according to Web Content Guidelines checkpoints:

1.1 Provide a text equivalent for every non-text element [Priority 1]

Refer also to WCAG checkpoint 9.1 and WCAG checkpoint 13.10.

Techniques for WCAG checkpoint 1.1 xt for84.019 -13.2 Td(HTML )Tj ET Q 6 w 2007.5 520

HTML

- Use a format such as SMIL which allows for the inclusion and synchronization of equivalent tracks

XML

- Use SMIL timing to synchronize equivalents

1.5 Until user agents

content.

SVG, XHTML

- Where SMIL animation is used, prompt the author to ensure that

## HTML

Use the same interface for defining areas of client- and server-side maps, and produce the image as client-side where possible

11.1 Use W3C technologies when they are available and appropriate for a task and use the latest versions when supported. [Priority 2]

Techniques for WCAG checkpoint 11.1

Raster images (PNG, JPEG, GIF)







General:

Ask the author to identify the language of any document. Provide a mechanism for setting a default.

5.1 For data tables, identify row and column headers. [Priority 1]

Techniques for WCAG checkpoint 5.1

5.2 For data tables that have two or more logical levels of row or column headers, use markup to associate data cells and header cells. [Priority 1]

For example, in HTML, use THEAD, TFOOT, and TBODY to group rows, COL and COLGROUP to group columns, and the "axis", "scope", and "headers" attributes to describe more complex relationships, including data cells.

Techniques for WCAG checkpoint 5.2

13.2 Td(532)Tj 15.29 0 Td(Do not, use tables for layout, unless the table makes sense when linearized)Tj -15.29 -13.2 Tdlinearized O th

s, provide linearized versions, and offer a link to the table for replacement. An example follows.)Tj 0 -13.2 Td( link fPrmk the tableFor ara replacement.Anr examplr tol,)Tj 0 -1

3.81cj 9649. 124163.81 952.5 2451.5 1952.5 2436.32 cj 952.5 24215.3 9649. 12408..81 980 2408..81cj 99(5.9 2408..812007



5.6 Provide abbreviations for header labels. [Priority 3]

Techniques for WCAG checkpoint 5.6

HTML

Prompt for an abbreviated form of each table header (th)

6.1

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

Techniques for WCAG checkpoint 7.1

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

Techniques for WCAG checkpoint 7.2

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

Refer also to WCAG guideline 8.

Techniques for WCAG checkpoint 7.3

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

Techniques for WCAG checkpoint 7.5

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

Refer also to WCAG guideline 6.

Techniques for WCAG checkpoint 8.1

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

Refer also to WCAG checkpoint 1.1, WCAG checkpoint 1.2, and WCAG checkpoint 1.5.

Techniques for WCAG checkpoint 9.1

HTML

where regions are not easily defined, ask the author to provide information that can be used to generate a form-based input

links have been focussed.

9.5 Provide keyboard shortcuts to important links (including those in client-side image maps), form controls, and groups of form controls.

[Priority 3]

Techniques for WCAG checkpoint 9.5

HTML

Ask authors to specify an accesskey for links that appear common

12.2

frame. Default text presented in the prompt could use the `title` defined for the document referenced in the `src`



transcript with prepackaged movies. Refer also to checkpoint 3.4.

- Use formats that allow for accessible annotation to be included in the files, such as SMIL, PNG, and SVG.
- Provide long descriptions, and associated text files with appropriate text equivalent in clip-art collections.
- Provide video description files with prepackaged video.
- Provide text caption files for prepackaged audio, or video with auditory track(s).
- Including pre-written descriptions for all multimedia files (e.g., clip-art) packaged with the tool will save authors time and effort, cause a significant number of professionally written descriptions to circulate on the Web,



information to be automatically retrieved whenever the author selected one of the packaged objects for insertion. An important benefit of the system would be the ease of adding a keyword search capability that would allow efficient location of multimedia based on its alternative information.



content view to make it clear that they are necessary. This will make it obvious when they are missing.



errors in content according to the Web Content Accessibility Guidelines. A draft of that document is available [AUTO-TOOL] .





•



part of the process. Any examples of code should give either block content with text links, or `AREA` elements that all have relevant `alt` attribute values.

- When an author creates an HTML frameset, suggest the links from the navigation bar (and perhaps the content of the "first page") as the content for the `NOFRAMES` element.
- **Sample:** In Amaya some accessibility features are part of relevant dialogs. Others, such as `longdesc` and `title` attributes must be separately generated

*Checkpoints:*

6.1 Document all features that promote the production of accessible content.

[Priority 1] (Checkpoint 6.1)

- Ensure that accessibility solutions are present in all help text descriptions of markup practices (e.g., HTML `IMG` elements should appear with an `alt` )



*Checkpoints:*

7.1 Use all applicable operating system and accessibility standards and conventions (Priority 1 for standards and conventions that are essential to accessibility; Priority 2 for those that are important to accessibility; Priority 3 for those that are beneficial to accessibility). (Checkpoint 7.1)

The techniques for this checkpoint include references to checklists and guidelines for a number of platforms and to general guidelines for accessible applications.

guidelines for application accessibility are

instance, the

Software

U

WCAG 1.0 [1]

systems that are d



Tools" [ACCESS-AWARE] , J. Richards.

2. **Reference:**

## *Input Device Independence*



7.4 Ensure that the editing view allows navigation via the structure of the document in an accessible fashion. [Priority 1] (Checkpoint 7.4)

- Some tools, such as those used to translate from one format to another, do not have an editing view.
- Allow the author to navigate via an "outline" or "structure" of the document being edited. This is particularly important for people who are using a slow interface such as a small braille device, or speech output, or a single switch

7.6 Allow the author to search within editing views . [Priority 2] (Checkpoint 7.6)

- Search functions are already present in almost every text an8ns to searcerns 1 i0 -13.2ing

[Relative Priority]

Note: Accessibility problems should be detected automatically where possible. Where this is not possible, the tool may need to **prompt** the author to make decisions or to manually check for certain types of problems.



## 3.3. Types of “Prompting”

All authoring tools will have ways of conveying information to users and collecting information in return. These methods vary according to factors such as the design of the tool and the user interface conventions for its platform. The following is a relatively generic overview of how these methods can be used for accessibility prompting. Keep in mind that these categories may overlap. For example, an intrusive alert may contain a prompt edit field.

### 3.1.1 Prompts

Prompts are basically requests for information. On most GUI platforms, prompts take the form of dialog boxes that request information from the user. The author answers the requests by setting or modifying control values (i.e. typing text in a textbox or selecting a checkbox). Prompts are relatively unintrusive because they are often displayed at the user's request. For example, when the user has chosen to save a



### 3.1.2 3.1.2 *Highlighting:*


Conformance with Checkpoint 5.2 may be reinforced by visually highlighting accessibility features with colour, icons, underlining, etc. For example, in Allaire's

**Tag Editor - INPUT**

Language (HTML 4.0)	Events (HTML 4.0)	VTML (Wizards)
INPUT tag	HTML 4.0	Style Sheets/Accessibility (HTML 4.0)

**Style Sheets**

Class:  ID:

Style:  

**Accessibility**

Access Key:  Tab Index:  This attribute is supported only by MSIE 4+ (not valid for Hidden)

Title:

Generate Label tag

Label text:



For a radio button or checkbox its caption (if defined) takes precedence.

**Notes:**

- If no ID is defined, an implicit LABEL tag will be generated.
- If a LABEL tag is generated, any access key will be defined on LABEL rather than the field except for a radio button or checkbox.
- MSIE4+ (only) supports LABEL but explicit labels only - specify an ID!

Quote all attributes (HTML)     Single quotes on attributes (HTML)

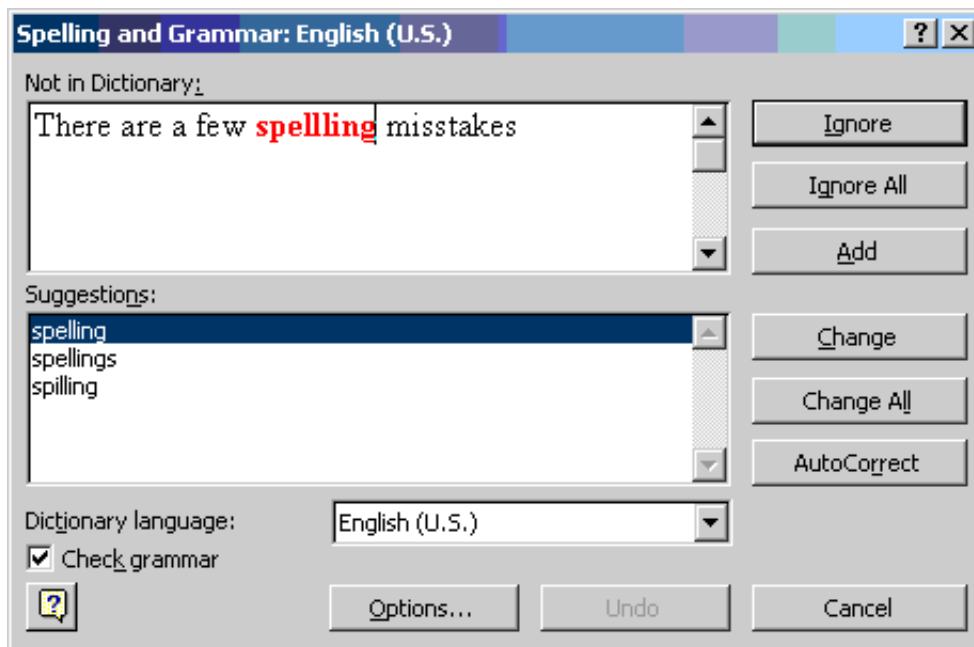
Output on single line

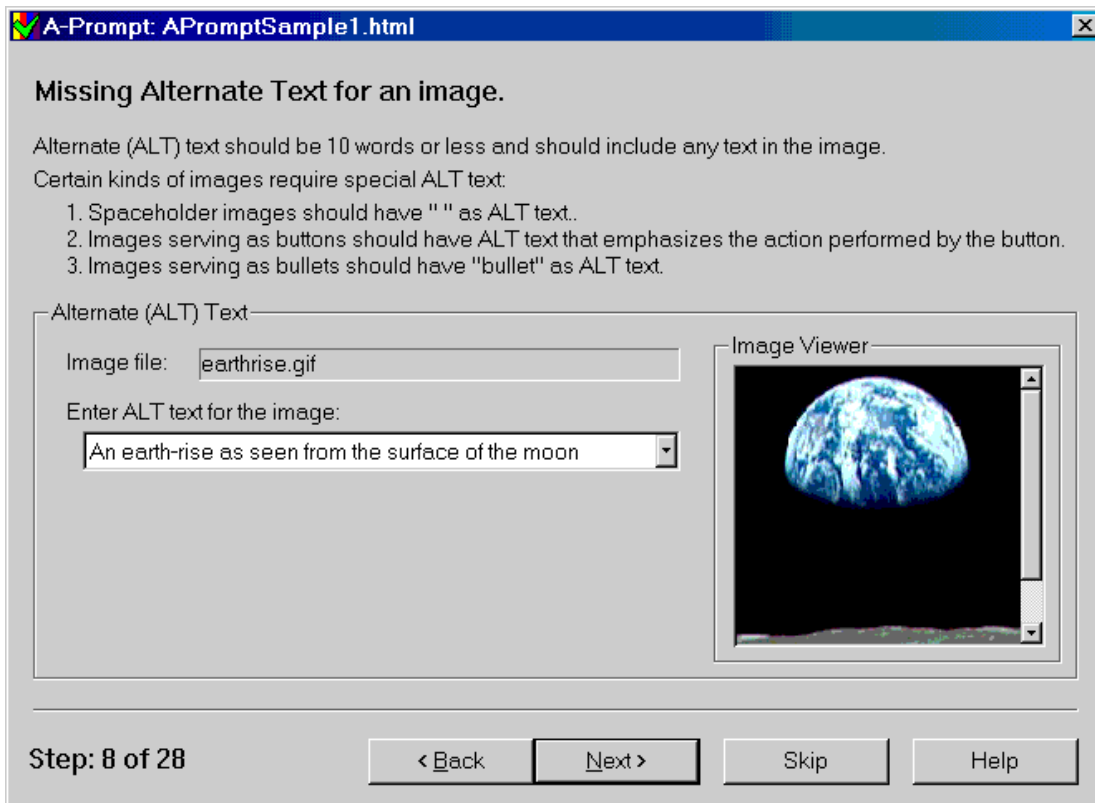
HELP  

OK Cancel

### 3.1.4 4.1.4 Sequential Prompts:

In some cases, authors may benefit from the sequential presentation of a number of prompts. This technique usually takes the form of a wizard or a checker. In the case of a wizard, relatively complex interactions are broken down into a number of simple steps so that later steps can take into account information provided earlier. For example, in a wizard for creating a new document, the first step might be to choose a template, and the second step might be to choose a title. The wizard would then use the information from the first step to present the second step, and so on.





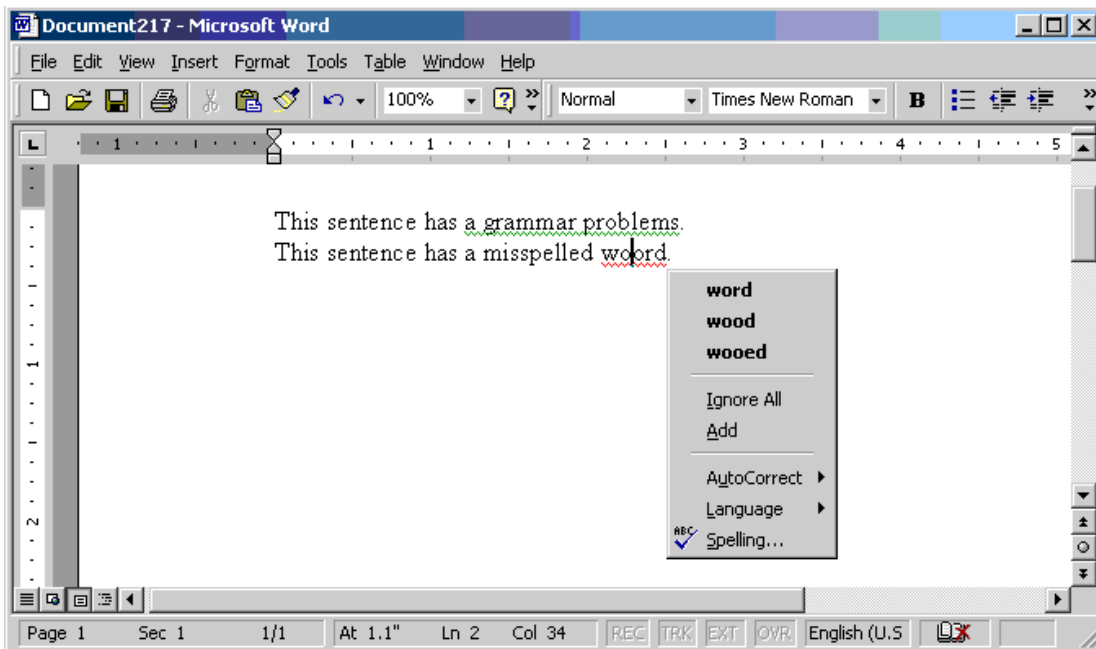
### 3.2 3.2 Alerts:

Alerts warn the author that there are problems that need to be addressed. The art of attracting the author's attention is a tricky issue. The way authors are alerted, prompted, or warned can influence their view of the tool and even their opinion of accessible authoring. Refer also to guideline 5.

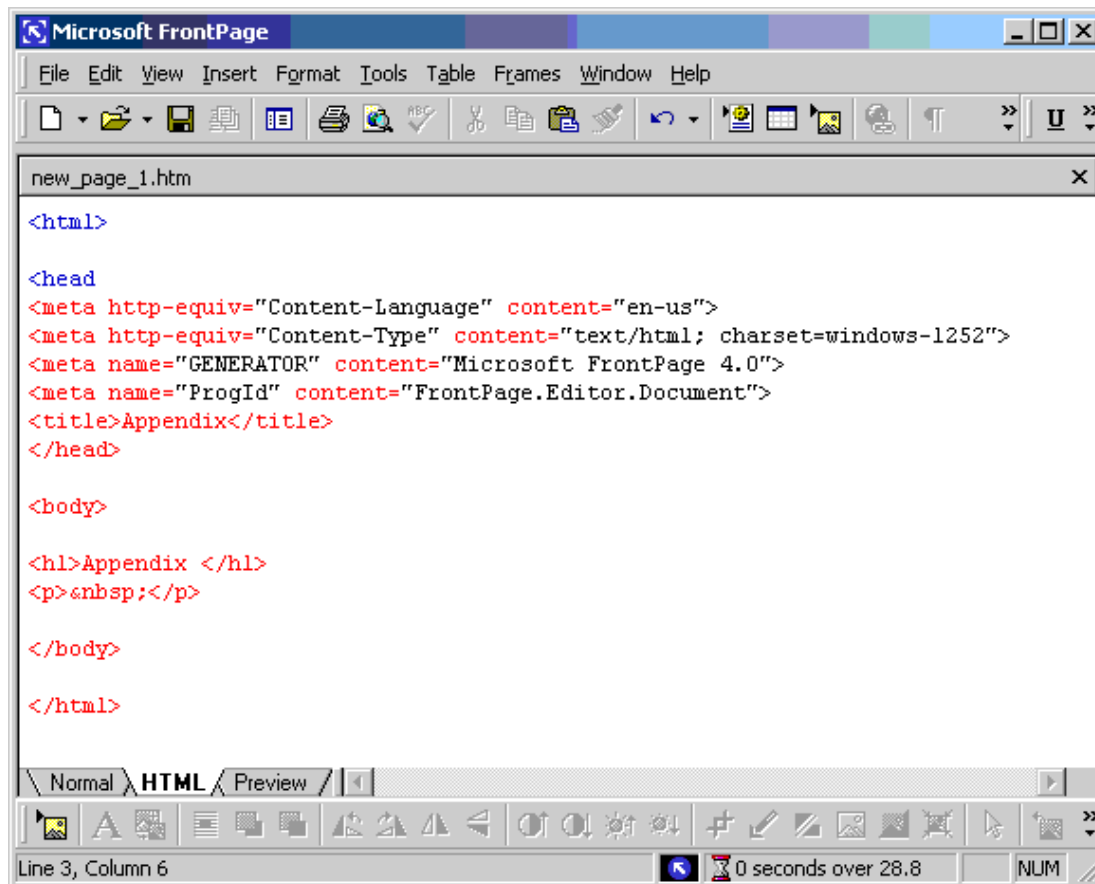
#### 3.2.1 3.2.1 Intrusive Alerts

Intrusive alerts are informative messages that interrupt the editing process for the author. For example, intrusive alerts are often presented when an author's action could cause a loss of data. Intrusive alerts allow problems to be brought to the author's attention immediately. However, authors may resent the constant delays and forced actions. Many people prefer to finish expressing an idea before returning to edit its format. The following screenshot shows an example of an intrusive alert that might be displayed if the author fails to enter Alt-text at an image insertion prompt.





Another Microsoft product, FrontPage 2000, uses unintrusive alerts in its HTML editing environment to indicate syntax errors. As the author types, the syntax is automatically checked. The author is allowed to make syntax errors, but the colour of the text signals that an error has been made.



The screenshot shows the Microsoft FrontPage 4.0 interface. The title bar reads "Microsoft FrontPage". The menu bar includes "File", "Edit", "View", "Insert", "Format", "Tools", "Table", "Frames", "Window", and "Help". The toolbar contains various icons for file operations and editing. The main window displays the HTML source code for a file named "new\_page\_1.htm". The code is as follows:

```
<html>

<head
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Appendix</title>
</head>

<body>

<h1>Appendix </h1>
<p>&nbsp;</p>

</body>

</html>
```

At the bottom of the window, there is a status bar showing "Line 3, Column 6", a timer "0 seconds over 28.8", and a "NUM" button.

In the context of the Authoring Tool guidelines, such unintrusive alert techniques could be used to indicate which parts of a document or site contain accessibility problems. This will inform the author about the type and number of errors without interrupting their editing process.

## 4 Glossary of Terms and Definitions

### **Accessibility** (Also: **Accessible**)

Within these guidelines, "accessible Web content" and "accessible authoring



challenging, since author perceptions of alerts, prompts, and warnings can influence opinions of the tool and even of accessible authoring.

An **Unintrusive Alert** is an alert such as an icon, underlining, or gentle sound that can be presented to the author without necessitating immediate action. For example, in some word processors misspelled text is highlighted without forcing the author to make immediate corrections. These alerts allow authors to continue editing with the knowledge that problems will be easy to identify at a later time. However, authors may become annoyed at the extra formatting or may choose to ignore the alerts altogether.

An **Interruptive Alert** is an informative message that interrupts the editing process for the author. For example, interruptive alerts are often presented when an author's action could cause a loss of data. Interruptive alerts allow problems to be brought to the author's attention immediately. However, authors may resent the constant delays and forced actions. Many people prefer to finish expressing an idea before returning to edit its format.

**Alternative Information** (Also: **Equivalent Alternative**)

Content is "equivalent" to other content when both fulfill essentially the same function or purpose upon presentation to the user. Equivalent alternatives play an important role in accessible authoring practices since certain types of content may not be accessible to all users (e.g., video, images, audio, etc.). Authors are encouraged to provide text equivalents for non-text content since text may be rendered as synthesized speech for individuals who have visual or learning disabilities, as braille for individuals who are blind, or as graphical text for individuals who are deaf or do not have a disability. For more information about equivalent alternatives, please refer to the Web Content Accessibility Guidelines WCAG 1.0 [WCAG10] .

Text equivalents for still images can be short ("Site Map Link") or long (e.g., "Figure 4 shows that the population of bacteria doubled ai inpulatiedidoubl2p8 0 -13.2 Td(an itw foy f may nimay Guo the t

***Auditory Description***

An "auditory description" provides information about actions, body language, graphics, and scene changes in a video. Auditory descriptions are commonly



### ***Inform***

To "inform" is to make the author aware of an event or situation through a visual, audio, prompt, sound, flash, or other means.

### ***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.

### ***Markup Language***

Authors encode information using a "markup language" such as HTML [HTML4], SVG [SVG], or MathML [MATHML].

### ***Multimedia Authoring Tool***

A "multimedia authoring tool" is software that facilitates integration of diverse media elements into an comprehensive presentation format. Multimedia includes video, audio, images, animations, simulations, and other interactive components.

### ***Presentation Markup***

"Presentation markup" is markup language that encodes information about the

desired presentation on a browser or other device. For example, the following code (desires a presentation should be used (or -SVG 21 p

Presentation

to be rendered by the user agent. This may differ from the element's structural content. For some elements, internal data may be rendered (e.g.,

***User-Configurable***

## 6 References

For the latest version of any W3C specification please consult the list of W3C Technical Reports at <http://www.w3.org/TR>.

[ACCESS-AWARE]

"The Three-tions of Accessibility-Aware HTML Authoring Tools," J. Richards.

[AMAYA]

Amaya, dev

WYSIWYG-s

specification

<http://www.w>

eds., 12 May 1998. This CSS2 Recommendation is <http://www.w3.org/TR/1998/REC-CSS2-19980512>. The latest version of CSS2 is available at <http://www.w3.org/TR/REC-CSS2>.

**[CSS2-ACCESS]**

"Accessibility Features of CSS," I. Jacobs and J. Brewer, eds., 4 August 1999. This W3C Note is <http://www.w3.org/1999/08/NOTE-CSS-access-19990804>. The latest version of Accessibility Features of CSS is available at <http://www.w3.org/TR/CSS-access>.

**[ED-DEPT]**

"Requirements for Accessible Software Design," US Department of Education, version 1.1 March 6, 1997.

**[EITACC]**

"EITACC Desktop Software standards," Electronic Information Technology Access Advisory (EITACC) Committee.

**[HTML-XML-VALIDATOR]**

The W3C HTML Validation Service validates HTML and XHTML markup.

**[HTML4]**

" /R12r.01 Recommendation," D. Raggett, A. Le Hors, and I. Jacobs, eds., 24 December 1999. This /R12r.01 Recommendation is <http://www.w3.org/TR/1999/REC-html401-19991224>. The latest version of /R12 4 is available at <http://www.w3.org/TR/html4>.

**[HTML4-ACCESS]**

"WAI Resources: /R12r.0 Accessibility Improvements," I. Jacobs, J. Brewer, and D. Dardailler, eds. This document describes accessibility features in /R1 r.0.

**[IBM-ACCESS]**

"Software Accessibility," IBM Special Needs Systems.

**[ICCCM]**

"The Inter-Client communication conventions manual." A protocol for communication between clients in the X Window system.

**[ICE-RAP]**

"An ICE Rendezvous Mechanism for X Window System Clients," W. Walker. A description of how to use the ICE and RAP protocols for X Window clients.

**[JAVA-ACCESS]**

"



MathML 1.0

**[TRACE-REF]**

"Application Software Design Guidelines," compiled by G. Vanderheiden. A thorough reference work.

**[UAAG10]**

"User Agent Accessibility Guidelines," J. Gunderson and I. Jacobs, eds. The latest version of the User Agent Accessibility Guidelines is available at <http://www.w3.org/WAI/UA/2> StTtidelines

Level Double-A conformance icon, W3C-WAI Web Content Accessibility Guidelines 1.0