Designing Universally Accessible Web Resources

for People with Disabilities

Jon Gunderson, Ph.D.
Division of Rehabilitation – Education Services
College of Applied Life Studies
University of Illinois at Urbana/Champaign
1207 S. Oak Street, Champaign, IL 61820
E-mail: jongund@uiuc.edu
### Outline

**Introduction**
- Myths of the Web .......................................................... 1
- Digital Divide .............................................................. 2

**Alternative Views of the Web**
- Keyboard Support ............................................................ 5
- Access to Text Descriptions .................................................. 7
- User Styling of text .............................................................. 13
- Speech Browsing ............................................................... 17

**Web Design Guidelines**
- W3C Web Content Accessibility Guidelines .......................... 20
- United States Section 508 Requirements ............................. 29

**Evaluation and Repair Tools**
- HTML Validation .............................................................. 33
- Evaluation Tools ............................................................... 33
- Evaluation and Repair .......................................................... 35
- Limitations of Current Authoring Tools .............................. 36
- PPT Accessibility Tool ......................................................... 37

**Accessible Repair or Universal Design?**

**Laws and Regulations**

**Conclusion**

**Bibliography**

**Internet Resources on Universal Design and the Web**

**List of Figures**
Abstract

The design of universally accessible web resources improves access to web resources for all users, including people with disabilities. The basis of universal design is to provide users with options and control over how they view information on the web. Universal design easily adapts to user capabilities or the capabilities of the technologies they are using. Users with visual impairments can style the text on the page to be easier to read. Able-bodied users using PDA or speech based telephone browsers can access content just as easily as someone viewing the information in a graphical rendering since the structure of the information is separated from the style of rendering. Even graphical renderings are enhanced since web resources adapt to the dimensions and resolutions of the users display settings. Universally designed resources can be as graphically rich as other web resources, but use different technologies to create the graphical effects.
Glossary

ADA: United States American with Disabilities Act of 1990 that guarantees the rights of people with disabilities to have access to public spaces, services and employment.

ATAG: W3C Authoring Tool Accessibility Guidelines provide authoring tool developers with information on how to support the creation of accessible web content and be more accessible to people with disabilities.

Text Equivalent: A text description associated with non-text content like images, audio and video.

ALT Text: The short description associated with an image on the web.

CSS (Cascading Style Sheet): Cascading Style Sheets is a W3C technology designed to style HTML and XML content for rendering in a browser. The specifications currently

Disability: A visual, hearing, muscular, learning or mental impairment that substantially limits one or more of the major life activities of an individual.

HTML Structure: Using the structural markup capabilities of HTML to indicate the relationships between information in a web resource.

Keyboard Shortcuts: The ability to use keyboard commands to control software.

LONGDESC: A URI to a detailed description of an image and is an attribute of the IMG element.

Section 508: United States Section 508 rules and regulations are designed for use by federal agencies to provide access to services by citizens and accommodations to employees with disabilities. In December of 2000 the Electronic and Information Technology Accessibility Standards included web accessibility requirements.

Screen Magnifier: A software program that magnifies text and graphics, and controls the colors on a graphical computer system

Screen Reader: A software program that is used by people who are blind to have the elements on a computer screen read to them through synthesized speech or refreshable Braille display.

UAAG: W3C User Agent Accessibility Guidelines provide information to developers of browsers and multi-media players on how to make their technologies more accessible.

Universal Design: The design of resources to adapt to the needs and capabilities of a wide range of users, including people with disabilities.
WAI: W3C Web Accessibility Initiative is a program of the W3C to promote the accessibility of the web to people with disabilities through education, design guidelines and review of web technologies for accessibility features.

WCAG: W3C Web Content Accessibility Guidelines provide information to web content authors on how to create accessible web materials.
Introduction

Tim Berners Lee developed the first HTML Web browser/editor in 1990 to enable scientists at the CERN particle physics lab in Switzerland to share electronic documents on a wide range of computing systems. At the heart of the design of HTML technologies is the concept of interoperability, the ability of providing and receiving electronic documents using public standards on wide variety of computing equipment. The use of public standards creates an environment where developers on a wide range of computing systems can develop tools for creating, serving and viewing electronic documents using the standards. In the beginning the focus was on the information and users typically had a wide range of choices and control over the rendering of web documents. Authors were not very interested in controlling the rendering of HTML and indeed the HTML markup language was limited their control.

Myths of the Web

As the web was commercialized through the introduction of graphical browsers (NCSA Mosaic, Netscape Navigator and Microsoft Internet Explorer) in the mid to late 1990s there was a fundamental change in the relationship between the control authors had over rendering style of web resources and the users ability to control rendering. There are many reasons for this shift, but the result is that the vast majority web authors developing commercial content for the web, primarily think of the web as a graphical medium. At the same time the most popular commercial browser developers provided users with fewer and fewer options for adjusting the rendering of web resources to a point where most web users today do not even know they have any control over the rendering of web content. Users too have developed a belief of the web as a graphical medium. This has
lead to the inaccessible design of web resources by web developers who increasingly create resources that only support graphical renderings and the use of the pointing devices (mouse) to interact with dynamic content. An example of this belief is seen in may web authors narrowing their view of interoperability to having web pages visually appear the same in both Netscape Navigator 4.7 and Internet Explorer (CITA Surveys, 2001a and 2001b). This approach leads developers to the use of images and complex table layouts for styling and positioning text and images, giving users little opportunity to access the content in non-graphical renderings like text or speech.

**Digital Divide**

The divide between people with visual impairments and able bodied web users was investigated by Pernice-Coyne and Nielsen (2001). They found that people who use screen magnification technologies could only complete web tasks about 25% of the time and people using speech output could only complete web tasks about 12.5% of the time. When compared to the able-bodied control group performance of completing tasks about 75% of the time, it is clear that current web design is creating tremendous barriers to people with disabilities. When the visually impaired and blind did complete tasks they took the about twice as much time and visited twice as many web resources as the control group. This indicates that web resources are not providing information on the structure or organization of the information on a web page that can be used by people with disabilities to efficiently identify and find information on the web resource.

It is estimated that in 1997 approximately 48 million Americans over the age of 15 years old have some type of disability and that about 17 million identified themselves as having
a severe disability (US Census Bureau, 2001). As people age the percentage of people with disabilities increases from 1.6 percent for people between the ages of 15 and 24 year old, to 5 percent for people between the ages of 15 and 64 years old, and then triples to 17 percent for people over the age of 65. So a major part of the financial argument for designing universally accessible web resources is designing web resources to deal with an increasingly aging population, and the economic power and productive capacities they bring to our nations economy. Kay (2000) found this barrier in the use of computer technology by people with disabilities. People with disabilities own computers at half the rate as the general population and use the Internet about a fourth of the rate as the general population. While there are many factors that influence computer ownership and Internet use, but probably one of the most critical is how the technology is designed to be inclusive of the needs of people with disabilities. Before concrete ramps and cut cuts were built-in to the physical structures of our society people with disabilities often had few places to go and were invisible. In the same way electronic ramps and curb cuts need to be built-in to our electronic infrastructure before we will see the wide spread presence of people with disabilities on the Internet.

**Alternative Views of the Web**

For the web to become more accessible to people with disabilities web authors need to understand that people will be viewing their resources in many different ways: including graphical, text and speech renderings. This section outlines the technologies people with disabilities use to access the web. This includes the rendering options of popular web browsers and specialized technologies designed specifically for people with disabilities.
The W3C User Agent Accessibility Guidelines (Jacobs et. al., 2002) outline the types of features browsers and multimedia players need to provide in order for people with disabilities to be able to access web content. One of the primary requirements is the ability to support the keyboard. People with many types of disabilities for various reasons can only use the keyboard to control the browser. Functions that are not available through keyboard commands will not be available to people with disabilities. People with disabilities also need to be able to select what types of content they want to view. For example, people who cannot see images, benefit more from text descriptions of the images. They would configure the current browsers to render the text description of an image in place of the graphical image. Other types of control include the ability to control the styling of text font characteristics, and the foreground and background colors when the text is rendered. People with visual impairments often need to use sans-serif fonts, larger text size and specific color combinations to make the text readable.

Automatic behaviors supported by many graphical browsers that can be disorienting, like authors automatically generating new windows. New windows are disorienting because the user is not expecting a new window to open when the follow a link. Usually the new window is given focus and when the user tries to use the back function to reorient themselves the document doesn’t change, since the new window doesn’t inherit the history of the primary window. While this is a usability problem for all users it impacts people with disabilities more since they many times do not have good information about the choices available to them or are oriented to the other open windows.

People who are blind cannot use the computer screen at all and use synthetic speech and refreshable Braille displays to access web content. Speech navigation and browsing is
much different than graphical browsing since the user only is able to view the content in a 
linearly. When using speech it is important to provide markup that allows users to skip to 
important structures like headers, navigation bars and links. Otherwise users need to read 
the entire document to understand the information available in the document.

**Keyboard Support**

Applications written for Microsoft Windows typically have very good keyboard support 
than applications written for Apple Macintosh and the various flavors of the Unix X- 
Windows systems. This generalization is also applicable to browsers. Browsers like 
Netscape Navigator (http://www.netscape.com), Internet Explorer 
(http://www.microsoft.com/ie) and Operasoft Opera (http://www.opera.com) all have 
keyboard shortcuts for the following functions:

- Next link or form control
- Previous link or form control
- Select link
- Move focus to next frame
- Reload content (Refresh)
- Stop loading content
- Move to previous resource in history
- Move to next resource in history
- Show next page of content
- Show previous page of content
- Next frame
• Menu bar options

• Navigation and setting of controls in dialog boxes

Opera provides additional keyboard commands that allow the user to navigate the structural elements of web resources. This includes individual functions to navigate by headers (H1-H6), form controls and HTML element-by-element. This type of function allows users to more efficiently identify the main topics of a web page, since they don’t need to rely on the styling the author used for the header to identify the text as a major topic. On pages with a large number of links it can be rather tedious to navigate to a specific link using the simple next link function available in Internet Explorer and Netscape Navigator. Opera allows the user to navigate past large numbers of links (if headers are used properly by the author) to the header closest to the link they want to select and then use the next link function from this position. Opera has a second function for navigating to links. The user can use a keyboard commands to view the list of links in the document and can use various keyboard commands to view and select individual links, including using the letter keys to move to the links that start with that letter. This type of function helps people with minimal range of motion use their physical ability more efficiently and people with visual impairments to have more options for searching and selecting a link. The keyboard shortcuts for a browser or multimedia player can typically be found in the help system (Figure 1).
Access to Text Descriptions

One of the most important configuration options is the ability to render text descriptions for images. HTML has two attributes of the IMG element that can be used to provide text descriptions, the ALT attribute for short descriptions and the LONGDESC attribute for providing a link to a longer description. Most graphical browsers render ALT text content in place of an image when the browser is configured to not render images, but the quality of the rendering varies considerably among current browser technology. One of the major issues with rendering text descriptions the differences in space required to render the text descriptions. Often text descriptions require more space than the original image, requiring a re-flow of content to accommodate the text descriptions be rendered. When images are used for spacing and positioning this can often create a distorted
rendering of text, making it more difficult for the user to understand the content relationships.

Currently the major graphical browsers Opera 6.1, Internet Explorer 6.0 and Netscape 6.2 do not fully support access to the text descriptions for all images only a subset or under special conditions. The HTML IMG element is the most popular way authors include images in web pages, but other elements including AREA and IMEPUT can have ALT attribute content. The IMG element includes an ALT attribute and the LONGDESC attribute for associating text descriptions with images. Table 1 shows the capabilities of various browsers in rendering ALT text descriptions.

<table>
<thead>
<tr>
<th>Browser</th>
<th>Operating System</th>
<th>Render ALT Text for IMG element</th>
<th>Render ALT Text for AREA element</th>
<th>User Styling of ALT text</th>
<th>Link to LONGDESC URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opera 6.01</td>
<td>Windows 98/2000/XP, Macintosh OS9 and Unix</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Internet Explorer</td>
<td>Windows 98/2000/XP</td>
<td>Yes, except when scripts</td>
<td>No</td>
<td>Limited</td>
<td>No</td>
</tr>
</tbody>
</table>
6.0  dynamically change the image source attribute

<table>
<thead>
<tr>
<th>Internet Explorer 5.5</th>
<th>Macintosh OS9</th>
<th>Yes, unless images are cached</th>
<th>No</th>
<th>No</th>
<th>No</th>
</tr>
</thead>
</table>

| Netscape 6.2 | Windows 98/2000/XP, Macintosh OS9 and unix | No | No | No | Yes, through context menu that is only accessible with mouse commands |

Table 1: Browser capabilities of rendering ALT text and provide a link to LONGDESC URI

Figures 2, 3 and 4 show the ALT text rendering of the same web page for Opera 6.1, Internet Explorer 6.0 and Netscape Navigator 6.2 respectively. Opera renders the ALT text and has extensive styling capabilities for the ALT text. Internet Explorer for Windows renders that ALT text and limits the ability of the user to style the ALT text. Netscape Navigator does not render ALT text when rendering of images are turned off. The text content of the ALT attribute is designed to provide a short text description of an image. The LONGDESC attribute provides a URI to a web resource that will provide a
more detailed description of the image. For example, if the image was a chart of what favors of ice cream people prefer at a certain ice cream store, the LONGDESC could point to a web page with a text table that representation of the preferences of ice cream. Opera has a very good implementation of rendering ALT text, since it provides the user with extensive control over styling the ALT text. Other implementations, including Internet Explorer, provide various levels of access to ALT text to no access in the case of Netscape 6.2. Access to the ALT text has much better implementation than access to the LONGDESC attribute. It is ironic that the one browser listed, Netscape 6.2, that provides access to the LONGDESC URI does not provide access to the ALT text description. The table shows that no browser provides complete access to ALT text descriptions limiting the types of content some users will have access to using these browsers.
Figure 2: View of ALT text rendering in Opera

Awards Convocation honors students, faculty, alumni

Forty-three students, twelve faculty members, and eight alumni were honored at the 38th College of Engineering Honor Awards Convocation, held Friday, April 19, 2002, in the Illini Union. New student awards in 2002 were the William R. Schowalter Award, created by Engineering Council upon the retirement of Dean Schowalter in 2001, and the Ben Jay Rosenthal Award, established in 2002 in memory of Mr. Rosenthal, BSIE ’63, by his family and the College of Engineering. The second Schowalter Award... Read more >>

News: Remodeled Mechanical Engineering Lab is...
Figure 3: View of ALT text rendering in Internet Explorer
The `<INPUT>` element of type “image” and `<AREA>` element that defines the clickable areas on an image MAP also have ALT attributes. Currently none of the major graphical browsers support the rendering of ALT text associated with the AREA element in any configuration. The lack of support makes the links of the image MAP elements inaccessible to people with disabilities. Authors should always provide redundant text links for both server side and client side image MAPs.

**User Styling of text**

People with visual impairments and learning disabilities that effect reading need to control the font characteristics, font size, and foreground and background colors of text.
In early graphical browsers, like NCSA Mosaic, this was a built-in feature and the user could completely configure the default style sheet used for rendering HTML. Current browsers vary widely in their ability to allow the user to control the rendering of text. Table 2 shows the capabilities of several popular browsers in allowing the user to control the rendering of text.

![Figure 5: Opera 6.0 settings for setting the rendering style information used in author and user modes](image)

The W3C Cascading Style Sheet (CSS) technology was designed to address many of the author and user styling issues of separating the structure of a web resource and the styling for a particular rendering. The advantage to developers in using style sheet technology is that a single style sheet can be used to control the rendering of any number of web pages,
making it easier for web masters to change the look and feel of their website without having to individually edit pages or elements. One of the most powerful aspects of the CSS specifications is user style sheets. The W3C realized that users need control over rendering and the specification includes the concept of user style sheets overriding author supplied style sheets. Opera has actually implemented the concept of user and author styling and provides a very concrete interface for users to select author and user styling preferences (Figure 2). Opera also provides a one step command (clickable icon and single key press) for the user to switch between author styling and user styling of web content. This is a very useful feature not only for user, but also web developers. Web authors can easily switch between their designs and a high contrast styling that might be used by someone with a visual impairment. The high contrast setting helps them to check if their web design will work for someone with a severe visual impairment or using portable technologies like a PDA or speech browsers that do have the same rendering characteristics as a graphical browser. Microsoft Internet explorer implements user style sheets, but does not allow the user to completely ignore style sheets supplied by the author. Table 2 shows the capabilities of various browsers in allow users to control author supplied styles and to apply their own style sheets.

<table>
<thead>
<tr>
<th>Browser</th>
<th>Operating System</th>
<th>Ignore Author Font Sizes, Font Style and colors</th>
<th>Adjust default Font Sizes, Font Style and colors</th>
<th>Ignore Author Style Sheets</th>
<th>Add user Style Sheet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software</td>
<td>Operating System</td>
<td>Font Size Control</td>
<td>Font Size Control Notes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------</td>
<td>-------------------</td>
<td>----------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Opera 6.01</td>
<td>Windows 98/2000/XP, Macintosh OS9 and Unix</td>
<td>Yes, limited font size control except through style sheets</td>
<td>Yes, limited to fonts and colors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet Explorer 6.0</td>
<td>Windows 98/2000/XP</td>
<td>Yes, limited font size control except through style sheets</td>
<td>Yes, but only in cascade with author style sheets (i.e. cannot use user style sheets when author style sheets are turned on)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netscape 6.2</td>
<td>Windows 98/2000/XP, Macintosh OS9 and Unix</td>
<td>Yes, limited font size control except through style sheets</td>
<td>Limited to fonts and colors</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Speech Browsing

Speech browsing is a fundamentally different way of accessing web information. In a graphical rendering the author often uses spatial relationships to group information on the screen and the users passively scan the screen to identify the grouping of information. But a speech rendering is temporal and requires the user to issue commands to direct the browser to read and reread content. The user could issue a command to speak the entire content of a web page, but in general this would be pretty inefficient way for the user to locate information they are interested in. It is the equivalent of a sighted user reading the entire contents of a web page before they started looking for links or other groupings of information on the page. Most able-bodied users scan the screen for highlighted text and other visual cues to identify the groups of information in the web page. In a well designed web page the author has intentionally created cues to help users focus their attention on information the author thinks is important. The same is true in speech browsing. If authors include structural markup, users using any technology can style that structure to highlight the information to the user by way of speech, text or graphical renderings.
Speech browsers like IBM Home Page Reader (http://www.ibm.com/able) and Freedom Scientifics JAWS (http://www.freedomscientific.com) screen reader have features for users to navigate HTML structural information. For example, they can navigate to elements marked as headers, links, form controls and to table data cells. These functions only work when the author uses HTML header and other markup correctly in their web resources. Figure 6 shows the read menu in IBM Home Page Reader (HPR) which highlights the different ways that HPR can be used to navigate through content. By providing users access to the structured markup it is easier for users to find the main groups of content without reading the entire document. Speech can be styled to indicate
different types of elements. In IBM Home Page Readers the reading voice for links is styled as a female voice and while non-link content is styled as a male voice for static text (voice can be configured to other settings by the user). Form controls that have explicit text labels associated with them can have their labels announced when the form control receives focus.

Many web pages do not contain structural markup. This forces speech browsers to look for implied structure if they want to offer the user more than just a linear reading of the document. In the example of form controls, speech browsers may try to calculate the relationships between the form control and text around the form control to determine the label for the control. This is problematic since the guess can be wrong and instead of helping the user, the user maybe confused over the purpose of the control. If images do not have ALT text descriptions, current speech browsers may use the file name of the image in hope that it may contain some useful information about the purpose and the content of the image. These approaches may help accessibility when the calculations are correct. When calculations are not correct they can seriously hurt accessibility by increasing the confusion and misinformation to users which results in users taking more time to explore the resource or improperly completing the task they are trying to complete.

**Web Design Guidelines**
The W3C Web Content Accessibility Guidelines (WCAG, 1999) and the United States Government Section 508 regulations for Electronic and Information Technology Accessibility Standards (Sect508, 2001) are the most widely used web design
requirements for designing or repairing web resources to be more accessible. The W3C Guidelines were designed from the perspective of the needs of people with disabilities and was a public and consensus based process. The section 508 web requirements were based on a need of the United States Federal government to define web accessibility standards that they felt were achievable and verifiable given the current state disability access to the web and the limitations of industry understanding and ability to create content that was accessible to people with disabilities. The section 508 requirements were loosely based on the Priority 1 requirements of the W3C Guidelines, so the requirements are similar.

**W3C Web Content Accessibility Guidelines**

A detailed review of the W3C requirements (WCAG, 1999) is beyond the scope of this chapter, since there are 64 checkpoints (requirements) organized into 14 guidelines with each requirement having an associated priority. The priority associated with each requirement corresponds to the relative importance of satisfying the requirement to the needs of people with disabilities. This section will highlight the organization of the guidelines and the major accessibility themes.

**WCAG Priorities and Conformance**

The following are the definitions of the priorities used in WCAG to identify how important a particular requirement is for people with disabilities to access content.

**Priority 1:** One or more groups will find it *impossible* to access information in the document if this requirement is not satisfied.
Priority 2: One or more groups will find it difficult to access information in the document if this requirement is not satisfied.

Priority 3: One or more groups will find it somewhat difficult to access information in the document if this requirement is not satisfied.

An author of a web document can claim conformance to the guidelines at one of the priority levels when they satisfy all the requirements at the desired priority level and the preceding levels. An author can claim Single A conformance to WCAG when all applicable Priority 1 requirements for their document are satisfied, Double A conformance when all Priority 1 and 2 requirements are satisfied and Triple-A conformance when all Priority 1, 2 and 3 requirements are satisfied. The publication of a conformance claim is useful to promote accessible design and provide users with expectations on the accessibility of the resource.

**WCAG Guidelines**

The guidelines are organized into logical groupings of requirements. The document was developed on the notion of HTML and CSS technologies, but the accessibility requirements where intentionally designed to be more generic to allow the requirements to be applied to other W3C and non-W3C technologies as they became available. The WCAG requirements do not have the specificity of requirements for HTML that are defined in the Section 508 web accessibility guidelines. The W3C guidelines do have an associated techniques document (WCAG Techniques, 2001) that does provide design
examples for HTML, CSS and other technologies. The techniques document is designed to be informative and authors are not required to use the techniques outlined in the techniques document to satisfy a particular WCAG requirement.

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

Images need to have text descriptions for people with visual impairments and some types of visual processing learning disabilities to understand the purpose and the content the image conveys in the document. Some images may require longer descriptions to convey the same information to the user. For example images of a famous painting or a satellite photograph would need a longer description if the author uses the image to convey important information in the document. An image of a numerical chart or table would need to have a text table of the same information available. Audio resources with speech need text transcriptions, and videos need to be captioned and have text descriptions of action.

**Guideline 2: Don't rely on color alone.**

Some web designers encode information in color. There are many people with visual impairments (including people with color blindness) that cannot see the information coded as a color. In this case their needs to be another means to convey the information. For example if the color of a score on a test was used to indicate the letter grade associated with the score or there were directions on the web page to press the red button. These are examples of information based on color. To correct this problem the actual
letter grade should be included with the scores and the text label of the button should be referred to instead of the color of the button.

**Guideline 3: Use markup and style sheets and do so properly.**

One of the main problems with current web design is that authors use graphical styling to encode the structure of the document. Instead of using the HTML H1 element to indicate the main topic of a document, authors use the FONT element to style the text for the main topic or an image of pre-styled text. So users who cannot use the author styling and apply their own styling or are using a speech rendering will not be able to identify the main topics of the document. When using HTML the proper way to indicate structure is to use the HTML elements like H1-H6, LABEL, CAPTION, TH, MAP and the list elements to properly indicate the structural between elements of information in the document and use Cascading Style Sheets to style the elements for different types of visual effects.

Validating HTML markup is also important to make sure that documents meet the requirements of the HTML language, which insures that documents can be rendered on any HTML compatible browser. Many times authors or authoring tools use proprietary features of a particular browser or create invalid markup that can only be rendered when using the HTML repair features of one or two browsers. This limits the choices users have for accessing the content. Many times authors are not even aware that they have created content that can only be rendered in one or two browsers. Creating valid HTML markup will become more important as the web matures and XML becomes more widely supported. Browser developers will want to focus there energies on exploiting the
capabilities of XML and not repairing invalid HTML markup. Slowly these repair
teatures will disappear from browsers, like in the move from Netscape 4.7 to Netscape
6.2.

Guideline 4: Clarify natural language usage

This requirement is critical for speech browsers, since the only way a speech browser can
identify the language reliably is when the author adds language information to the
document. In HTML every element can include a LANG attribute to indicate the
language of the content of the element. For example a document that is primary English
should use <HTML LANG="en"> in the beginning of the document. If the author uses a
French quotation, the container element for the quote should include a LANG="fr".

Guideline 5: Create tables that transform gracefully.

Most of the HTML table markup used on the web is for graphical positioning. This ia a
potential problem for speech renderings that read information in document order. Table
formatting that puts connected information out of document order can be confusing to
speech users or people who use technologies that do not render table markup (i.e. Lynx
browser). If tables are used for layout they should be as simple as possible and should be
tested with a speech browser, a text only browser or a graphical browser like Opera that
can be configured to ignore table markup to verify that the linear rendering makes sense
when the table markup is removed.

Guideline 6: Ensure that pages featuring new technologies transform
gracefully.
Technologies like Macromedia Flash, Adobe Acrobat and XML technologies like MathML, SVG and WAP have varying degrees of disability access solutions, so when using technologies like these you will need to determine the current extent to which the technology support users with disabilities. Many times technologies will not be able to meet the needs of major disability groups and alternatives will need to be created that provide a more accessible version of the information. This type of information redundancy should not be necessarily considered a problem, but as an opportunity to provide information in more than one form that provides everyone with an opportunity to use information that is in a form that is most useful to them and their needs.

**Guideline 7: Ensure user control of time-sensitive content changes.**

People who need extra time to read information or who have physical impairments that slow their response time need to be able to have additional control over time sensitive information. Providing mechanisms for the user to receive extra time to respond to a prompt is important and should be an option on pages with time sensitive input. In secure environments it would be sufficient to allow user settings or configuration options to provide the extended response information throughout the system.

**Guideline 8: Ensure direct accessibility of embedded user interfaces.**

Embedded technologies like Java and Active-X need to not only be compatible with assistive technologies, but also have built-in accessibility features. This may require adding additional controls to the control to allow the user to style text and other objects presented through the embedded interface; or to provide an option for the user to style the interface based on the users operating system style preferences. Keyboard support is
important in the design of embedded user interfaces and the user needs to be able to control automated behaviors. Many times technologies will not be able to meet the needs of major disability groups and an alternative will need to be created that provides a more accessible means for people to access the information accessed through the imbedded interface.

**Guideline 9: Design for device-independence.**

One of the major problems for dynamic web content is that designers only include support for pointer devices. It is important to use device independent events to allow users to interact with the content using the widest number of input devices possible, including only the use of the keyboard. When device independent event handlers are not available make sure that you at least support the keyboard and mouse pointer for all the functionalities of your dynamic content.

**Guideline 10: Use interim solutions.**

There are gaps between what browser and assistive current technologies can offer for accessibility and what specifications provide as accessibility features. This is by nature a dynamic requirement, so the requirements in this section should fade as technologies become obsolete.

**Guideline 11: Use W3C technologies and guidelines.**

The use of W3C technologies are recommend since current W3C technologies have been reviewed for accessibility features and support open and interoperable standards. This means that people using W3C technologies can support users with disabilities and also provide users with more choices to access content. Technologies like Adobe Acrobat and
Macromedia Flash are adding accessibility features, but their features are based on their own internal definition of accessibility requirements and users are still limited to the capabilities of their players for rendering information accessibly. In contrast technologies like HTML, CSS and SMIL are supported by many developers and give the user more choices in accessing content.

**Guideline 12: Provide context and orientation information.**

One of the primary problems in current website design is the lack of information that can be used by non-graphical renderings to identify the structure and the relationships of information on the page. Many web authors view the web as primarily a graphical medium and use graphical methods to encode the structure of the document. These graphical techniques don’t translate the structural relationships very well to text and speech renderings. Often the graphical techniques used to indicate structure actually cause information to be distorted in non-graphical renderings. Non-graphical renderings typically use document order as the means to render information. If table markup is used to position information for a graphical rendering the document order often separates connected pieces of information making the non-graphical rendering confusing.

**Guideline 13: Provide clear navigation mechanisms.**

Navigation is an important issue, especially for accessing information in web sites, documents that have a large number of links or large structured documents. Some examples of how markup can be used to improve navigation in HTML include:

1. The text associate with a link to indicate the destination of the link.
2. The use of markup to provide users with a means to skip over repetitive navigation links
3. Use of the MAP element to indicate a collection of related links
4. Use of the LABEL element to indicate the purpose of a form control

**Guideline 14: Ensure that documents are clear and simple.**

A requirement to use clear and simple language and layout is often very subjective and is often linked more to usability than for disability access. But since many people with cognitive disabilities may have language impairments it is important to carefully review the terms and organization of web resources to make the resources as easy read as possible. Carefully consider the types of people who will be using the web resources and their tasks and interests.

It is important to look at create web resources from the perspective of users and not from managers and other employees. One of the biggest problems in website design is that many people design the website to meet the needs of the designer, the desires of the sponsors of the website or the manager of the organization the website is representing. This often results in designs that meet the sponsor’s needs, but all to often do not meet the very different needs of intended users. People within the organization usually understand procedures and relationships that users coming to the website do not. This often results in too much information on the main page, jargon that is unfamiliar to the users and the expenditure of resources to create visual effects that increase visual esthetics of the web site, but do little to help the user to understand and complete tasks on the web resource.
The web is an electronic environment that has the unique capability of allowing people to select how they want to access the information or if the web resources provides them with clear options. This allows the creation of a range of web pages that can offer different interfaces based on the needs and the capabilities of the user.

**WCAG 2.0 Development**

WCAG 2.0 is currently under development and this will supercede the current WCAG 1.0 requirements. For more information on the current status of WCAG 2.0 or to participate in the groups activities go to their home page: http://www.w3.org/WAI/GL/

**United States Section 508 Requirements**

The section 508 web Electronic and Information Technology Accessibility Standards (sect508, 2000) developed by the Access Board of the United States Federal Government includes accessibility requirements for all electronic machinery, computers and software used by the federal government. Other regulations of the Access Board of the access board have been used as the basis of the Americans with Disabilities Act (1990) technical requirements.

The web accessibility requirements of Section 508 are based on only the W3C Web Content Guideline Priority 1 requirements. Therefore section 508 requirements are considered a minimum accessibility requirement, basically insuring that a web resource is not impossible for a person with a disability to access. One of the organizational differences between WCAG and Section 508 is that section 508 is specific to HTML and CSS technologies. Section 508 also has an additional requirement on functional
performance. The functional performance requirement is a general requirement for all the requirements of Section 508, but applies also to the web requirements. The web requirements are minimal for accessibility and authors are encouraged to consider more accessible design features than 508 requires.

The following are the section 508 requirements for web content with comments related to specific WCAG 1.0 checkpoint requirements.

a. A text equivalent for every non-text element shall be provided (e.g., via "alt", "longdesc", or in element content). (compatible with WCAG Checkpoint 1.1)

b. Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation. (compatible with UAAG Checkpoint 1.4)

c. Web pages shall be designed so that all information conveyed with color is also available without color, for example from context or markup. (compatible with UAAG Checkpoint 2.1)

d. Documents shall be organized so they are readable without requiring an associated style sheet. (compatible with UAAG Checkpoint 6.1)

e. Redundant text links shall be provided for each active region of a server-side image map. (compatible with UAAG Checkpoint 1.2)

f. Client-side image maps shall be provided instead of server-side image maps except where the regions cannot be defined with an available geometric shape. (compatible with UAAG Checkpoint 9.1)
g. Row and column headers shall be identified for data tables. (*compatible with UAAG Checkpoint 5.1*)

h. Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers. (*compatible with UAAG Checkpoint 5.2*)

i. Frames shall be titled with text that facilitates frame identification and navigation. (*compatible with UAAG Checkpoint 12.1*)

j. Pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz. (*compatible with UAAG Checkpoint 7.1*)

k. A text-only page, with equivalent information or functionality, shall be provided to make a web site comply with the provisions of this part, when compliance cannot be accomplished in any other way. The content of the text-only page shall be updated whenever the primary page changes. (*compatible with UAAG Checkpoint 11.4*)

l. When pages utilize scripting languages to display content, or to create interface elements, the information provided by the script shall be identified with functional text that can be read by assistive technology. (*no WCAG 1.0 equivalent*)

m. When a web page requires that an applet, plug-in or other application be present on the client system to interpret page content, the page must provide a link to a plug-in or applet that complies with §1194.21(a) through (l).
This is important user functionality that is part of the W3C User Agent Accessibility Guidelines 1.0, instead of WCAG 1.0.

n. When electronic forms are designed to be completed on-line, the form shall allow people using assistive technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues. (requires more than UAAG Checkpoint 10.2 and 12.4)

o. A method shall be provided that permits users to skip repetitive navigation links. (no WCAG 1.0 equivalent)

Skipping repetitive navigation links is consider important usability feature to help user skip over repetitive navigation bars and advertisements to get to the main content of a document faster.

p. When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required. (no WCAG 1.0 equivalent)

This is important user functionality, and this problem is addressed in the W3C User Agent Accessibility Guidelines 1.0, instead of WCAG 1.0 by the W3C.

**Evaluation and Repair Tools**
The Section 508 requirements and W3C WCAG guidelines can be rather tedious to use and to many authors the terminology used in the guidelines is unfamiliar. Automated analysis and repair tools have been developed to help authors identify the accessibility problems in their websites.
HTML Validation

HTML validation is not usually considered an accessibility check, but valid HTML documents help accessibility in two ways. First valid documents are more reliably rendered in a wider range of technologies, including specialized technologies for people with disabilities. Since people with severe disabilities often need to use less popular or custom technologies to access web information. The second way is that HTML has requirements that support accessibility. For examples the inclusion of an ALT attribute (short text description of the image) for IMG and AREA elements is a required for a document to be valid, which is the most common accessibility problem on the web.

Figure 7 shows the HTML validator service of the W3C (http://validator.w3.org/).

Figure 7: Image of the web based HTML validation service offered by the W3C.

Evaluation Tools

The first generation of automated accessibility evaluation tools is exemplified by CASTs Bobby website and software (http://www.cast.org/bobby). Bobby can provide an evaluation of a web resource based on either the Section 508 or WCAG requirements. Bobby analyzes the markup used by the author and reports on known or potential accessibility problems. Known problems are easily identified when markup is missing, for example the ALT attribute on an IMG element. Other problems like the accessibility of scripting or the use of multiple languages in a web document cannot be easily
determined through a computational analysis and require the author to manually determine if there is a problem. In this case Bobby indicates to the user a manual check is needed to determine accessibility. An example report from the web based version of Bobby can be viewed in Figure 8.

Figure 8: Image of an evaluation report generated by the Bobby Accessibility Evaluation web resource.

The advantage of using Bobby or similar tool is that the author only needs to deal with the accessibility of the markup they are actually using. For example, if scripts are not used in a document the report does not include any information to check the accessibility of scripts. This helps the author to focus their attention on the problems of their particular website design, essentially a custom set of guidelines for their design style.

One of the limitations of tools like Bobby is when they check for the presence of markup, like ALT attributes for IMG elements, they can’t check to see if the ALT text content really represents the use or information the image conveys. For example many authoring practices and automated authoring tools use the name of the image as the ALT text for the image, just to satisfy HTML validation requirements. While some tools will flag this as a potential problem, others just assume the ALT text is useful and do not report the problem. Tools like Bobby usually generate a large number of manual checks, which leaves the author in the same position as using the guidelines directly of having to try to interpret the guidelines to determine if they satisfy the requirements. Although the
automation tool can dramatically reduce the scope of the manual checks to only the ones
the author needs to address.

**Evaluation and Repair**

The second generation of accessibility evaluation tools provide both evaluation and repair
services. In addition to identifying known or potential accessibility problems the second
generation tools help the author repair the document. For example if an image is missing
the ALT text for an IMG element the repair tool can prompt the user add the ALT text
within the tool and add it to the HTML markup. The ability to repair content directly in
the tool improves the efficiency of authors in repairing content, since they don’t need to
go back into their original authoring tool to make the repairs. A-Prompt in Figure 9 is an
example of a stand-alone tool that provides evaluation and repair services. Similar
evaluation and repair tools have been developed for HTML authoring tools. LIFT from
Usablenet ([http://www.usablenet.com](http://www.usablenet.com)) is an example of a tool that works within
Macromedia Dreamweaver and Microsoft FrontPage to help authors correct accessibility
problems within the authoring tool. Figure 10 shows an example of the LIFT extension
for Dreamweaver.

Figure 9 about here: figure09-a-prompt.bmp

Figure 9: A-Prompt evaluation and repair tool

Figure 10 about here: figure10-lift.bmp

INSERT IMAGE: figure10-lift.bmp
There are clear advantages to incorporating repair functions into accessibility automation tools, since authors receive additional guidance in repairing problems and they don’t need to keep switching back between the evaluation report and the authoring tool to make changes to the document. These types of tools assume that the user is developing a static web page and has access to the source markup. Web resources that are generated through server side scripts and databases do not benefit from these types of tools, because the HTML markup generated by user each time a user makes a request to the URI. Also not all HTML markup can be repaired through simple prompts (like missing ALT text) and will require more extensive revisions to the content than the repair tool can offer.

**Limitations of Current Authoring Tools**

The need for automated accessibility evaluation and repair tools indicates a severe weakness in current HTML authoring tools in helping authors intrinsically create accessible materials by default, rather than by exception. Ideally authoring tools should make it easier for people to create universally accessible web resources and guide them in the use web markup that supports accessibility. Currently authors need to have information on accessible design to create accessible content with most HTML editors. Some authoring tools actually impede the ability of the author to create accessible information. For example in Dreamweaver Version 4, the author cannot easily use the MAP element (typically, but not limited to use for image maps) to indicate a collection of related text links (i.e. a navigation bar). Dreamweaver warns the user that this is an invalid use of the markup (apparently Dreamweaver has been designed to only support
AREA elements in a MAP container) and the text links can only be hand edited into the code. But it is structural markup like MAP that is critical for making the next generation of web technologies more accessible. Dreamweaver in this case actually makes it harder to move to the next level of accessibility.

**PPT Accessibility Tool**

Not all authors use traditional HTML authoring tools to create content. For example in educational institutions the most popular web authoring tools used by instructors is Microsoft Office. Instructors can use the familiar features of Office products and save the content in an HTML like format. Unfortunately the default Save features of Office produce XML content that can typically only be viewed in Internet Explorer. Authors who are more inquisitive can change the default settings to publish as HTML that can be viewed on other browsers, but this requires additional skill and knowledge on the part of the author. In either case the content developed is not very accessible and in this case the author does not even have an option to repair the markup, since Office tools do not provide a means to even hand edit the resulting code.

While the current situation with Office is not very good for publishing accessible web content, there is tremendous potential with these types of tools to automatically generate accessible content. By using the application programming capabilities of these products new save options can be developed that by default create accessible content and can guide authors into adding additional accessibility information. An example of one type of tool is a tool that can convert Microsoft Power Point Slides into accessible HTML (http://www.rehab.uiuc.edu/ppt). It automatically creates two parallel linked versions of
HTML slides. One set of slides uses primarily text and CSS to provide a highly user customizable version of the slides and the other is the more traditional graphical version of the slides. Each version of the slide is linked to the other version so the user can easily move between a graphical view and a text view of the slides. This illustrates another important web and accessibility concept. Giving users the choice on how they want to view information. Unlike print materials which become more expensive or inconvenient to provide multiple views of the same information, there is little cost on the web. Users can therefore pick the view that works for them based on their own needs and the task they are trying to complete. The tool also prompts authors for additional information when needed for accessibility. But unlike current evaluation and repair tools the prompts are non-technical and ask the user for the information, hiding the HTML coding. Figure 11 shows the prompt for creating a text equivalent for an image. The user is asked about how the image is used in the presentation and then guides the author in creating a compatible text equivalent.

Figure 11: Power Point Web Accessibility Plug-in asking the user the purpose of an image in the presentation.

**Accessible Repair or Universal Design?**
The main characteristic of both Section 508 and W3C Web Content Accessibility Guidelines is that they organize their requirements on a model of an author having already prepared their web materials (for example an existing website for example) and the author is trying to repair the website to be more accessible. An alternative approach
to the design/repair model is based on universal design and making HTML design choices at the beginning of the design process or totally redesigning the web resource with more accessible HTML markup techniques. The primary philosophical change for most authors to move to a universal design model of web development is to move away from a graphical view of web development and towards an information view where the author has preferred rendering styles, but makes it easy for the user to adapt the rendering to their technology or personal needs and preferences. This includes mobile technologies like cell phones and PDAs using speech interaction techniques or character oriented displays similar to the technologies used by people with disabilities.

While a detailed discussion of universal design of web resources is beyond the scope of this chapter. The following is a summary of the general approach for the development of static HTML documents (no scripting effects):

1. Don’t use images to stylize text, instead use text with CSS styling (including creating visual button effects)
2. Use HTML header markup (H1-H6) to indicate new sections or major topics
3. Use HTML MAP element to indicate collections of related links (navigation bars)
4. Use HTML list markup to indicate ordered or unordered lists of information and use CSS list styling to customize bullets and numbering
5. Markup document language and language changes with the LANG attribute
6. Use HTML LABEL element to indicate the text labels associated with form controls
7. Use HTML TABLE markup sparingly for positioning elements on a page
8. Use CSS background color and image capabilities instead of images for creating background effects
9. Don’t use images or CSS absolute position features for positioning, instead use CSS margin and padding to position information within layout tables
10. Provide text equivalents for all non-text content (i.e. images, audio and video)
11. Use the TH element, SCOPE and HEADER attributes to indicate header cells in data tables
12. Use only valid HTML and CSS techniques, do not support proprietary extensions of any particular browser

When these techniques are used it makes the resource not only accessible to people with disabilities, but provides all users with more flexibility to access content.

**Laws and Regulations**
Many countries including the United States have been starting to legally require web accessibility for government and public websites (Thatcher et. al., 2002). In the United States the Section 508 Web Accessibility requirements apply to Federal web resources. There is a weak provision in the rule that any state receiving Technical Assistance funds (~$250,000/year) would also need to comply with the Section 508 requirements for state agencies. Since states were not involved in the design of the requirements, it is likely most states would give the money back to remove the requirements if they felt it was a burden. Under ADA and Section 504 the Rehabilitation Act of 1973 both call for “effective communication” of information and in a timely manner. Since the web is now
a major means of communication of information it would be hard to argue that providing “effective communication” of web information could be done through some other medium or technology. Therefore the main question is what standard should be used to determine if a web resource is accessible. At the minimum right now the Section 508 requirements would probably be considered the minimum in the United States since the Federal government has adopted them. But these could easily be shown to be not enough to provide effective communication. For example Section 508 does not have a requirement for including language information. Language markup is needed for web sites with more than one language for speech output systems to know when to switch to speaking another language. Without this most multi-language web sites are not accessible. These types of websites are commonly used in on-line language education courses. Additional litigation will probably need to occur before the legal requirements of accessibility are clearly understood.

**Conclusion**

The universal design of web content not only provides users with disabilities access to web content, but all users will have more choices and more control over the rendering of web content. Just like concrete curb cuts and ramps have benefited the general population in many ways, the electronic web accessibility curb cuts and ramps will benefit all web users.
Bibliography


CITA Surveys (2001a), Available at http://cita.rehab.uiuc.edu/survey/web-masters-survey-result.html, (Date of access: June 6, 2002)

CITA Surveys (2001b), Available at http://cita.rehab.uiuc.edu/survey/ADA-survey-result.html, (Date of access: June 6, 2002)


Internet Resources on Universal Design and the Web

Illinois Center for Accessible Instructional Design: http://cita.rehab.uiuc.edu

Trace Center: http://www.trace.wisc.edu

W3C Web Accessibility Initiative: http://www.w3.org/WAI

Access IT: http://www.washington.edu/accessit

Information Technology Technical Assistance and Training Center (ITTATC): http://www.ittatc.org/

National Center on Accessible Media: http://ncam.wgbh.org/

CAST: http://www.cast.org

Webable: http://www.webable.com

Usablenet: http://www.usablenet.com

Adobe: http://access.adobe.com

IBM: http://www.ibm/able

Microsoft: http://www.microsoft.com/enable

SUN: http://www.sun.com/access/
List of Figures

Figure 1: Information on Internet Explorer shortcut keys found in the Help system

Figure 2: View of ALT text rendering in Opera

Figure 3: View of ALT text rendering in Internet Explorer

Figure 4: View of ALT text rendering in Netscape Navigator

Figure 5: Opera author and user styling settings

Figure 6: IBM Home Pager read menu options

Figure 7: HTML Validator

Figure 8: Bobby Evaluation Tool

Figure 9: A-Prompt Evaluation Tool

Figure 10: LIFT Evaluation Tool

Figure 11: Power Point Accessibility Plug-in