

## **Accessible Design Guidelines**

Sarah Horton  
Dartmouth College

**Version 5**

November 19, 2002

The Web is a visual medium, so the biggest challenge for Web developers is designing pages that are accessible to users with visual disabilities. Thus, many of the guidelines detailed here have to do with developing pages that accommodate different requirements for viewing, and are a “good read” when read aloud by screen reader software. Unfortunately, there are inconsistencies with the way screen readers read Web pages, so some of the guidelines may or may not have a noticeable effect. It is good practice to follow the guidelines regardless since Web browsers and adaptive technologies like screen reader software will surely make use of the information in future versions.

Please note that this is a “living” document and not all guidelines and conventions are noted. I will continually update the guidelines to reflect new or newly acquired techniques for accessible design. The most current version of the document can be downloaded from:

<http://www.dartmouth.edu/~webteach/resources/download.html>

## GENERAL

**Favor clarity and consistency**

All users benefit from clear and consistent Web site design, but for some it is critical. With a lack of spatial cues and with radically different approaches to navigation that must be relearned at every site, it is far too easy to get disoriented or lost on the Web. For people with cognitive disabilities such as memory or learning disabilities, this difficulty is increased many-fold. Stick with simple language and navigation applied consistently throughout your site, and everyone will benefit.

**Supply text alternates for all *relevant* non-text elements**

Blind users are likely to be the people most affected by your Web design choices. Blind users most often use screen reader software, a speech-enabled browser, or a Braille display, and designing pages that are understandable when “read” by these devices is probably the biggest challenge for Web designers. Users with cognitive or learning disabilities also use screen reader software because having the text read aloud while reading along visually helps them understand the materials. These adaptive technologies can transform *text* into something that is accessible to people with different needs. But the Web is not only about text, and any non-text item on a Web page presents a potential barrier, particularly for visually disabled users. Provide alternative text for any non-text element.

**Provide equivalent alternates**

Sometimes designing accessible pages is simply a matter of telling the same story different ways. People with cognitive or learning disabilities may benefit from having audio descriptions available as well as text, so you could include “audio notes” to accompany your text content. You could supply images (with appropriate ALT-text) along with text for navigation and page content, which would also make your pages more accessible to users with cognitive or learning disabilities. Deaf or hearing-impaired users, as well as others, benefit from captions with video content. Provide an equivalent version whenever you include content in a modality that may not be accessible to all users.

**Use structural markup and style sheets**

With CSS-styled pages, users can easily apply personalized formatting to Web documents. Some browsers have a feature that allows users to override author-defined style sheets with their own style sheet. A page designed using red text against a green background, for example, presents a problem for people with red-green color blindness: the contrast between text and background may not be great enough for the text to be readable. If text color is set using <FONT COLOR> and headings are set using <FONT SIZE> and <B> for emphasis, the user-defined style sheet will have nothing to apply itself to (no paragraph or heading tags). If the colors are set via a style sheet, users can set their browser preferences to override your settings and can apply their own style sheet to the page instead. Use CSS-styled pages, so users can transform Web content into a format that meets their requirements for accessibility.

**Use flexible layouts for graceful transformation**

Many disabilities can be accommodated by standard browser software as long as the Web pages are flexible and can transform to meet the viewer's needs. For example, color-blind users can apply their own text and background color to a page to increase legibility, or low-vision users can enlarge the text to a size they can read comfortably. Some low-vision users read better with white or yellow text on a black background. Design pages that hold up to these transformations – that remain legible and navigable under different viewing conditions.

## TEXT

**Use logical markup**

Use tags that describe what your text *means*, not what you what it to *look* like. Without logical markup a screen reader will not be able to distinguish elements such as headings from body text, and will not pause or change tone accordingly. Use CSS to define the presentation aspects of your pages.

**Avoid** presentation markup

Example: <FONT SIZE>, <FONT FACE>, <B>, </>

**Use** logical markup

Example: <H1>, <H2>..., <EM>, <STRONG>, <CITE>

**Pay attention to punctuation**

Screen readers use punctuation cues to modulate the tone and measure of the reading, e.g., a colon will cause the screen reader to pause; a period will produce a cadence and pause. Be liberal with punctuation as these pauses and cadences greatly enhance the readability of your text. Without punctuation the reader continues without pause or change in inflection, so unpunctuated phrases run together into one long and jumbled sentence.

Also, avoid using text to convey visual information, such as using a “>” or “/” symbol as navigation to show the path to the user’s current location. The screen reader will read the symbol literally rather than interpret the visual meaning you wish to convey.

**Avoid** unpunctuated phrases

Example:

- Use logical markup
- Pay attention to punctuation
- Make lists explicit

Sounds like “use logical markup pay attention to punctuation make lists explicit”

**Use** punctuation to set off phrases where appropriate

Example:

- Use logical markup.
- Pay attention to punctuation.
- Make lists explicit.

Sounds like “Use logical markup. Pay attention to punctuation. Make lists explicit.”

**Avoid** using text to convey visual information

Example:

Home > Text > Do not use text as graphics

Sounds like “Home greater than Text greater than Do not use text as graphics”

**Make lists explicit**

Blind users do not have the benefit of context and visual cues when it comes to reading lists – they have no way of knowing where the list begins, how many items are in the list, where the list ends. You can aid blind users by constructing lists so they make sense without visual cues.

**Avoid** unannounced and unnumbered lists

Example:

Guidelines for developing accessible text are:

- Use logical markup
- Pay attention to punctuation
- Make lists explicit

Sounds like “Guidelines for developing accessible text are use logical markup pay attention to punctuation make lists explicit”

**Use** numbering and explanation

Example:

The following three guidelines are essential for developing accessible text:

1. Use logical markup.
2. Pay attention to punctuation.
3. Make lists explicit.

Sounds like “The following three guidelines are essential for developing accessible text: 1. Use logical markup. 2. Pay attention to punctuation. 3. Make lists explicit.”

**Identify acronyms, abbreviations, and language changes**

Screen readers can't do much with language changes unless they're identified as such. The software will just plow along trying to pronounce the word, which in many cases produces nonsense. Be sure to flag and spell out acronyms, abbreviations, and language changes. [?]

**Use** the ABBR element with the TITLE attribute for abbreviations

Example:

```
<ABBR TITLE="Street">St.</ABBR>
```

**Use** the ACRONYM element with the TITLE attribute for acronyms

Example:

```
<ACRONYM TITLE="Cascading Style Sheets">CSS</ACRONYM>
```

**Use** the LANG element for language changes

Example:

```
<SPAN LANG="es">jHola!</SPAN>
```

**Write descriptive link text**

Never use “click here” alone as link text. Often, users using screen reader software tab through documents to hear the available links on a page. If your links are all tagged as “[click here](#) for...” the user will hear “click here, click here, click here” without the context of the surrounding sentence or paragraph. The best practice for link titles is to write the sentence as you normally would and place the link anchor on the word or words that best describe the content that you are linking to.

**Avoid** “click here” link titles

Example:

[Click here](#) for the Web Content Accessibility Guidelines

[Click here](#) for the Accessible Design Guidelines

Sounds like “click here click here”

**Use** link titles that describe the linked content

Example:

[Web Content Accessibility Guidelines](#)

**Do not use color to convey meaning**

If you rely on color alone to achieve typographic emphasis, users who cannot distinguish the colors will miss the emphasis. To emphasize text – for example, in headers or key phrases within text – use other typographic devices such as varying the font size or weight instead of color.

**Avoid** colored text

Example:

```
<H3 STYLE="color: blue">Do not use color to convey meaning</H3>
```

**Use** typographic emphasis

```
<H3 STYLE="font-weight: bold">Do not use color to convey meaning</H3>
```

**Use high contrast colors for text and background**

Be sure there is sufficient contrast between the background and text on your page. The biggest determiner of contrast is the lightness of a color, so colors with contrasting lightness values (black and white, dark purple and light yellow) will produce the most contrast. Some users with low vision find it easier to read white or yellow text against a black background. The best way to provide for individual needs is to use style sheets to define your document's color settings (text, links, and background colors) so users can easily override your settings and apply their own custom style sheet.

**Avoid** adjacent hues with similar lightness and saturation values

**Use** dark colors from the bottom of the hue circle (blue, violet, purple, and red) with light colors from the top half of the hue circle (blue-green, green, yellow, and orange)

**Use** style sheets to designate colors

Example:

```
body {    background: white;
          color: black }
a:link {  color: #0000ff }
a:visited { color: #663399}
a:active { color: red }
a:hover { color: #000099; text-decoration: underline }
```

**Use scalable type**

Users cannot easily enlarge text that is set using absolute size values; for example, text sized using pixels. To ensure scalability, use relative units such as percentages or the em unit to control the typography – type size, margins and indents, leading – on the page.

**Avoid** fixed measurements such as pixels

**Use** relative units

Example:

```
p {      font-size: 1em;
          font-family: Verdana, Arial, Helvetica, sans-serif;
          line-height: 150% }
```

**Favor HTML text over graphic text**

Use text graphics sparingly, and always offer a text-only equivalent. Text rendered to graphic form is no longer text but image and cannot be manipulated – enlarged, colored – as plain text can.

## GRAPHICS

**Use ALT-text where appropriate**

ALT-text is an HTML fallback designed to bridge the gap between purely visual information and text. Without ALT-text many users cannot access Web sites, particularly when navigation is supplied via graphics alone, because the screen reader software will either skip over the image or say something unhelpful like “image” and move on. To write effective ALT-text you need to think about the meaning and purpose of the image – what the image “says” – and then convey it in a few words. When you include images like spacer graphics that should not be announced to the user be sure to include an empty ALT tag (ALT="") so screen reader software knows to skip over the image.

**Avoid** missing ALT text

**Avoid** unhelpful ALT text

Example:

```
<IMG SRC="banner.gif" ALT="site banner">
```

**Use** descriptive ALT text

Example:

```
<IMG SRC="banner.gif" ALT="Accessible Design Guidelines">
```

**Avoid** ALT-text for irrelevant images

Example:

```
<IMG SRC="spacer.gif" ALT="spacer graphic">
```

**Use** blank ALT-text for irrelevant images

Example:

```
<IMG SRC="spacer.gif" ALT="">
```

**Provide alternate text for graphic navigation**

Some screen reader software will read the ALT-text you assign to the clickable areas of your imagemaps, which means blind users can navigate using graphic navigation. You should also provide text-only links as an alternate navigation option because people with other types of vision impairments cannot customize graphic navigation.

**Avoid** missing ALT-text in image maps

Example:

```
<AREA SHAPE="rect" COORDS="0,0,137,27" HREF="index.html">
```

Sounds like “Map: index.html”

**Use** ALT-text in image maps

Example:

```
<AREA SHAPE="rect" COORDS="0,0,137,27" HREF="index.html" ALT="Accessible Design Home">
```

Sounds like “Accessible Design Home”

### **Describe images on a separate HTML page**

The HTML attribute LONGDESC is intended to take up where the ALT tag leaves off by allowing Web developers to supply a longer description for non-textual elements. Use LONGDESC to describe images that require more description than brief ALT-text.

#### **Use LONGDESC**

Example:

```
<IMG SRC="otters.jpg" ALT="Photograph of sea otters" LONGDESC="otters.html">
```

### MULTIMEDIA

#### **Provide alternates for any multimedia content**

There are several approaches to making multimedia content accessible to users with disabilities and each requires effort and expertise:

- For hearing impaired users, include a text description and transcript for any audio materials. You should synchronize text with video to provide captions for video materials.
- For visually impaired users, include a description for any visual materials, either as a separate audio file or as text.
- For users with reduced vision or cognitive difficulties, simplify the materials, for example, by reducing a video track to a series of still images synchronized with audio.

Providing a text transcript is fairly simple if you have a video script available. Alternatively you can transcribe the audio track using a tool like MAGpie (see *Tools*, below). Describing a video track is substantially more difficult, though if you are regularly developing video content, creating a video description should be part of your development process.

## FORMS

**Label form fields**

If you simply place a form field on a page and do not have a label associated with it, users who are read Web pages will have no way of knowing what the field is there for. Associate labels with their form fields by positioning them together on the page and by using the LABEL tag to associate the label text with its form field. Note that you need to include the ID attribute in your INPUT tags whenever you use the LABEL tag.

**Use** LABEL to associate fields with their labels

Example:

```
<LABEL FOR="firstname">First name: </LABEL><INPUT TYPE="text" NAME="firstname"
ID="firstname">
<LABEL FOR="lastname">Last name: <INPUT TYPE="text" NAME="lastname"
ID="lastname">
```

**Make tabbing order sequential**

The tabbing order is the order in which elements receive focus when the user presses the Tab key. Links and form elements receive focus. The default tabbing order is based on the order in which elements appear in the HTML code. You can explicitly set the tabbing order using the TABINDEX attribute.

**Use** TABINDEX to set a logical tabbing order

Example:

```
<INPUT TYPE="text" NAME="firstname" TABINDEX="1">
<INPUT TYPE="text" NAME="lastname" TABINDEX="2">
```

**Always include a submit button**

Many Web sites use the popular single-field search feature that allows users to type their query into the field and press Enter to submit. With screen reader software, pressing Enter in a field activates input into the field. This means that, without an explicit Submit button, users will not be able to submit their query. Always include a submit option, either as the standard form tag or as a custom image button. If you use a custom image for a button be sure to include appropriate ALT text.

**Use** a submit option

Example:

```
<INPUT TYPE="text" NAME="searchquery"><INPUT TYPE="submit" NAME="Search">
```

**Use** ALT-text for graphic buttons

Example:

```
<INPUT TYPE="image" NAME="submit" SRC="submit.gif" ALT="Submit">
```

**Provide an alternate to forms**

Web forms may present too many difficulties, so always provide an alternate means for submitting information or placing an order such as a phone (voice/TTY) and fax number, email address, and/or mailing address.

**Use access keys**

Access keys are keyboard shortcuts that you can assign to page elements such as input fields and links. These shortcuts improve keyboard navigation and can be a real timesaver for disabled users. Be sure not to override shortcuts that are already used by the browser or screen reader software (keys to avoid include F, E, C, V, G, A, H, left arrow, and right arrow). Note that Windows users use the Alt key and Macintosh users the Ctrl key to access HTML-defined keyboard shortcuts.

**Use** access keys to facilitate keyboard navigation

Example:

```
<LABEL for "searchquery">Search: </LABEL>  
<INPUT TYPE="text" ID="searchquery" NAME="searchquery" ACCESSKEY="s">  
<INPUT TYPE="submit" NAME="Search" >
```

## LAYOUT

**Provide “skip to main content” option**

The top of a Web page is prime real estate and is often used for advertising, navigation, site identity, and graphics. Specifically, navigation links at the top of the page can be very useful for sighted users who are browsing, but once you have found what you’re looking for, it is easy to skip directly to the main content. In other words, this is generally not an area you re-read on every page of a site – once you figure out the basic structure you home in on the content of the page. Users who have pages read to them have more difficulty skipping over irrelevant content. Screen reader software starts at the top of the page, which means users have to listen to the advertising, site identity, and navigation options *on every page*. Provide a link at the beginning of the page that moves the reader directly to the main content. You can do this with a text link or by using a transparent spacer graphic so that sighted users will not see the link.

**Use** a skip to main content device

Example:

```
<A HREF="#main"><IMG SRC="spacer.gif" ALT="Skip to main content"></A>
```

...

```
<A NAME="main"></A><H1>Main page content</H1>
```

**Do not use structural tags for visual effects**

Web designers have co-opted standard structural HTML tags to achieve layout and formatting effects, for example, using the BLOCKQUOTE to indent text. Pages that incorporate these workarounds can produce unexpected results when read aloud by screen reader software. Use structural tags appropriately.

**Avoid** using HTML tags for visual effects

Example:

```
<BLOCKQUOTE> Web designers have co-opted standard structural HTML tags to achieve layout and formatting effects.</BLOCKQUOTE>
```

Sounds like “Begin quotation Web designers have co-opted standard structural HTML tags to achieve layout and formatting effects. End quotation”

**Design flexible layouts**

Many Web users require special font sizes or color settings in order to access Web content. If you design pages that rely on certain environment constants in order to be legible, those users may not be able to view your pages. Design flexible layouts that transform gracefully under different viewing conditions.

**Avoid** narrow fixed-width layout tables that cannot accommodate large type

**Use** tables that adapt their contents

**Design tables that make sense when linearized**

Most screen reader software looks at the HTML code of Web pages and reads tables in a cell-by-cell, or *linearized*, fashion. This means that each cell of the table becomes a line or paragraph of text that is read in sequence. Make sure your content appears sequentially in your source file so that, for example, all navigation links are in the first cell and all content in the second. One way to test a layout is to save your page as text-only and then open the file and see if the page makes sense without the table formatting.

**Avoid** complex layouts that will not make sense when linearized

**Use** simple layout tables and keep related content in the same cell

**Use table headers and IDs or scope for data tables**

Since linearized tables are read in row-wise order, column headers become disassociated from their contents. This makes data tables particularly difficult to understand for those using screen reader software. Use the TH tag to identify column headers and use the ID or SCOPE attribute to link headings with their associated table contents.

**AVOID** confusing data tables

Example:

```
<TR><TD>Text</TD><TD>Graphics</TD><TD>Multimedia</TD></TR>
<TR><TD>Logical markup</TD><TD>ALT text</TD><TD>Alternatives</TD></TR>
```

Sounds like Text Graphics Multimedia Logical markup ALT text Alternatives

**Use** the TH tag with the ID and HEADERS attributes

Example:

```
<TR><TH ID="text">Text</TH><TH ID="graphics">Graphics</TH><TH
ID="multimedia">Multimedia</TH></TR>
<TR><TD HEADERS="Text">Logical markup</TD><TD HEADERS="Graphics">ALT
text</TD><TD HEADERS="Multimedia">Alternatives</TD></TR>
```

Sounds like "Text: Logical markup, Graphics: ALT text, Multimedia: Alternatives"

**Use** the TH tag with the SCOPE attribute

Example:

```
<TR><TH SCOPE="col">Text</TH><TH SCOPE="col">Graphics</TH><TH
SCOPE="col">Multimedia</TH></TR>
<TR><TD>Logical markup</TD><TD>ALT text</TD><TD>Alternatives</TD></TR>
```

Sounds like "Text: Logical markup, Graphics: ALT text, Multimedia: Alternatives"

**Use the SUMMARY attribute to define the purpose of tables**

The SUMMARY attribute of the TABLE tag is a non-displaying element that screen reader software can read aloud to announce the functional purpose of a table. This is particularly useful for layout tables, providing a means to communicate to the user that the table is for visual layout purposes only.

**Use** the SUMMARY attribute to label tables

Example:

```
<TABLE WIDTH="100%" CELLPADDING="0" CELLSPACING="0" BORDER="0"
SUMMARY="Page layout table">
```

**Use frame titles**

The visual relationship between the content in frames is usually obvious for users who can see: for example, navigation links in the left frame, content in the right. Without the benefit of visual cues, however, blind users will have difficulty orienting themselves in a frames-based layout. There are additional reasons to avoid a frames-based layout, but if you must use frames, be sure to include titles for the FRAME tags and a NOFRAMES alternative to navigating your site.

**Use** the TITLE attribute to label frames

Example:

```
<FRAMESET COLS="20%, 80%" TITLE="Accessible Design Guidelines">
<FRAME SRC="nav.html" TITLE="Navigation">
<FRAME SRC="general.html" TITLE="General Guidelines">
```

**Use** a NOFRAMES alternate

Example:

```
<NOFRAMES><A HREF="content.html" TITLE="Table of Contents">Accessible Design
Guidelines Table of Contents</A>
</NOFRAMES>
```

**Watch out for inline images**

A common layout approach is to align images so that the text of the paragraph runs around the image. The trouble is that a screen reader will read the ALT-text of the inline image right along with the paragraph text. In general you should avoid using inline images.

**Avoid** using the align attribute with images

Example:

```
<P>A common layout approach is to <IMG SRC="example.gif" ALT="Example of aligned
image" ALIGN="right"> align images so that the text...</P>
```

Sounds like “A common layout approach is to Example of aligned image align images so that the text...”

**ACCESSIBLE DESIGN CHECKLIST**

1. Turn off graphics. Make sure you can understand and navigate the site with only the supplied ALT-text.
2. Turn off style sheets. Make sure your pages are still readable without style sheet formatting.
3. Turn off features like JavaScript, frames, plug-ins, and scripting. Make sure your pages are still usable with these features disabled.
4. Set the text zoom to its maximum value. Make sure your text resizes, and that your page layout can accommodate large text.
5. Resize your browser window. Make sure the layout holds up to different window widths.
6. Navigate your site from the keyboard. Make sure you can access all navigation and form elements without using a mouse. Make sure you cycle through links in a logical order. Verify that the text of your links makes sense when read out of context.
7. Check your pages with monochrome settings. Make sure there is sufficient color contrast, particularly between text and background.
8. Check your pages using screen reader software like Home Page Reader (see *Tools*, below). Make sure the page makes sense when read aloud.
9. Save your pages as text-only from the browser. Make sure your pages make sense when read linearly.
10. Validate and preview your pages using the tools listed below under *Verifying Web pages*.

### RESOURCE LINKS

#### Verifying Web pages

Bobby: Accessibility validation service

<http://bobby.watchfire.com/>

Lynx Viewer: Lynx (text-only browser) simulator

<http://www.delorie.com/web/lynxview.html>

UsableNet: Accessibility validation service

<http://www.usablenet.com/>

Vischeck: Color-blindness simulator

<http://www.vischeck.com/>

WAVE: Accessibility validation service

[http://www.temple.edu/inst\\_disabilities/piat/wave/](http://www.temple.edu/inst_disabilities/piat/wave/)

W3C HTML Validation Service

<http://validator.w3.org/>

W3C CSS Validation Service

<http://jigsaw.w3.org/css-validator/>

#### Web design guidelines and tutorials

Accessible Design Guidelines

<http://www.dartmouth.edu/~webteach/resources/download.html>

Flash Accessibility

<http://www.macromedia.com/macromedia/accessibility/features/flash/>

How to Create Accessible Adobe PDF Files Booklet

<http://access.adobe.com/booklet.html>

IBM Accessibility Center

<http://www-3.ibm.com/able/>

Microsoft Accessibility

<http://www.microsoft.com/enable/>

Rich Media Resource Center

<http://ncam.wgbh.org/richmedia/>

WebAIM

<http://www.webaim.org/>

Web Accessibility Initiative (WAI)

<http://www.w3.org/WAI/Resources/>

Web Content Accessibility Guidelines 1.0.

<http://www.w3c.org/tr/wai-webcontent/wai-pageauth.html>

#### Tools

A-Prompt: Web accessibility validation software

<http://aprompt.snow.utoronto.ca/>

Betsie: Text-only generator

<http://www.bbc.co.uk/education/betsie/>

Home Page Reader: Speech-enabled browser

<http://www-3.ibm.com/able/hprtrial3.html>

MAGpie: Captioning and describing software

<http://ncam.wgbh.org/webaccess/magpie/index.html>

## BIBLIOGRAPHY

**Articles**

- Accessibility guidelines for electronic resources, *Library Technology Reports*, 37:4 July/August 2001. [see also [www.techsource.ala.org](http://www.techsource.ala.org)]
- Blaser, Art. 2001. Distance learning — boon or bane? *Ragged Edge Magazine* (September). <http://www.raggededgemagazine.com/0901/0901ft1.htm> (21 September 2001).
- Brewer, Judy, ed. 2001. *How people with disabilities use the Web*. <http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/> (4 December 2001).
- Chancellor's Office, California Community Colleges. 1999. *Distance education: Access guidelines for students with disabilities*. <http://www.htctu.fhda.edu/dlguidelines/final%20dl%20guidelines.htm> (9 April 2002).
- Coombs, Norman. Electronic ramp to success: Designing campus Web pages for users with disabilities. *Research Bulletin: Center for Applied Research* (19 March 2002). <http://www.educause.edu/asp/doclib/abstract.asp?ID=ERB0206>.
- Horton, S. Beauty is only screen deep. *Boxes and Arrows* 14 October 2002, [http://www.boxesandarrows.com/archives/beauty\\_is\\_only\\_screen\\_deep.php](http://www.boxesandarrows.com/archives/beauty_is_only_screen_deep.php).
- Horton, S. Making Web accessible to all. *The New York Times* 10 June 2002, C4. [See also <http://www.nytimes.com/2002/06/10/technology/10NECO.html>]
- Horton, Sarah. 2001. *Practical accessibility*. <http://www.dartmouth.edu/~webteach/articles/access.html> (9 April 2002).
- Mates, Barbara. 2000. Adaptive technology for the Internet: Making electronic resources accessible to all. <http://www.ala.org/editions/samplers/mates/>.
- Waddell, J.D., Cynthia D. 1998. *Applying the ADA to the Internet: A Web accessibility standard*. <http://www.rit.edu/~easi/law/weblaw1.htm> (26 February 2002).
- Waddell, J.D., Cynthia D. 1999. *The growing digital divide in access for people with disabilities: Overcoming barriers to participation*. <http://www.aasa.dshs.wa.gov/access/waddell.htm> (16 May 2001).

**Books**

- Clark, J. 2002. *Building accessible websites*. New Riders Publishing.
- Lynch, P., and S. Horton. 2002. *Web style guide: Basic design principles for creating Web sites*, 2nd ed. New Haven: Yale University Press. [See also <http://www.webstyleguide.com/>]
- Paciello, Michael G. 2000. *Web Accessibility for People with Disabilities*. Gilroy, Calif.: CMP Books. [See also <http://www.webable.com>]
- Slatin, J., and S. Rush. 2002. *Maximum accessibility: Making your web site more usable for everyone*. Addison Wesley Professional.
- Thatcher, Jim, et al. 2002. *Constructing Accessible Web Sites*. glasshaus.