Coding for Section 508 Compliance

Contents

List of Figures

List of Tables

Coding for Section 508 Compliance

A guide for making web pages more accessible.

1. Introduction: Goals of Section 508

Section 508 is part of the 1998 amendment to the Rehabilitation Act. It is designed to make technology used by federal government employees, or technology provided by the government for use by its constituents, accessible to people with disabilities. Those disabilities include visual problems such as blindness, poor eyesight, and color-blindness; hearing problems such as deafness and hearing loss; and motor disabilities which prevent a person from using a mouse.

In the broadest sense, 508 requirements ask you to think about this: How would people use your application if they were blind? Deaf? Unable to use a mouse? How can you design your application so that it is accessible by all?

In many cases, the answers are not difficult. In its continuing evolution of web-based technology, the W3C has consistently striven for accessibility on many levels. It has designed features such as alternative text, title attributes, captions, and summaries to enable you, as web-developers, to provide all your users, even those with disabilities, the opportunity to share the benefits of the web.

In the pages that follow, we will highlight some of the ways you can meet the goals of 508. In a short document like this one, however, we cannot cover all the possibilities or address every situation. Consequently, we have provided a section of additional useful readings after our discussion of each 508 requirement. Please refer to those sources for additional explanations or clarifications.

NOTE:

In the spirit of making content more accessible, all the correct code samples in this document follow XHTML 1.0 Strict guidelines. Although not required by 508 guidelines, XHTML is recommended for all web pages to make them more accessible to a variety of user agents.

2. Section 508 Guidelines

Section 508 consists of a number of parts. The Technical Standards include:

- Software Applications and Operating Systems (1194.21)
- Web-based Intranet and Internet Information and Applications (1194.22)
- Telecommunications Products (1194.23)
- Video or Multimedia Products (1194.24)
- Self Contained, Closed Products (1194.25)
- Desktop and Portable Computers (1194.26)

This document concerns itself with the requirements of section 1194.22, Web-based Intranet and Internet Information and Applications with only occasional glances at section 1194.24 Video and Multimedia Products since these kinds of products are sometimes used in the training modules that support web-based applications.

Each of the sections that follow consists of the following information:

- a statement of the actual 508 requirement
- a brief explanation of its intent
- a series of succinct recommendations for meeting that requirement
- a short list of additional readings

2.1 Providing Alternative Text

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

The primary purpose of this requirement is to provide sight-impaired users with a way of understanding the *pertinent* information conveyed through the visual images on the page. But *it also includes* information conveyed by sound that hearing-impaired users might not be able to hear. Examples of non-text-elements include images, graphs, buttons, icons, scanned text (i.e. an image of textual elements) and embedded or streaming audio or video.

- Provide an **`alt`** attribute for all images, image maps, **`<input type="image" />`** elements, and applets--no exceptions.

NOTE:

The **`<applet>`** element is now deprecated, and should, when possible, be replaced by the **`<object>`** element.

- Provide concise but meaningful alternative text for images, image maps, **`<input type="image">`** elements, and applets with real content, especially if the images contain information needed by the user for either navigational or informational purposes.
  - If an image contains words on it, the **`alt`** text should, at minimum, contain those words, e.g. “Submit.” or “Offer good through June 1, 2006.”
  - If an image has no words, but conveys information, the **`alt`** text should convey the function of that image as accurately as possible, e.g. “Woman demonstrating correct ergonomic position of desk chair as described in text.”

- If an image conveys complex information such as a chart, graph, or map, use the **`alt`** attribute to provide a summary of the purpose of the image e.g. “Chart showing changes in distribution of wealth among economic groups: 1970 to 2004.”). Use the **`longdesc`** attribute or an explicit link adjacent to the image for more detailed information describing the actual content.

- Punctuate alternative text according to the standard rules of grammar. This means *always* putting a period "." (or other appropriate end punctuation) at the end of each **`alt`** text string so that screen readers know they have arrived at the end of one item.
- Provide a blank **`alt`** attribute (alt="")for images without content, for instance, spacer gifs.
- Provide a blank **`alt`** attribute for icons or other decorative graphics when the icon or graphic (an arrow icon, for instance) is part of a link with its own descriptive text.
- Provide minimally descriptive **`alt`** text for icons or other decorative graphics when the icon is decorative and *not* included as part of a link, for instance, **`alt="PDF Icon."`** or **`alt="Arrow Icon."`**
- Provide complete **`alt`** text for the destination of an image or icon when the icon is separate from the link but is an anchor for the same destination. This can occur, for instance, when an arrow icon precedes a link and is also “hot.” In this case, the alt text should be something like this: **`<img src="arrow.gif" alt="Long Distance Plans: Arrow Icon." />`** The destination is described before the image itself so that the user can make a decision sooner.
- Avoid links and images without descriptive or contextual content, like “Click Here,” or “Learn More,” or “Go.” Users with screen readers often use a special function that grabs and presents all the links on a page together. If the link itself has no content or context, the user will have no idea of where it goes. If you must use such links, be sure to include contextual information in the **`alt`** or **`title`** attribute as appropriate. For a “Learn More” button associated with the AT&T 5-Cent Plan, for instance, the right **`alt`** attribute might be “AT&T 5-Cent Plan: Learn More.”
- If you include PDF documents in your site, make sure that they have been created from a digital source file (rather than being scanned) and can be read by a screen reader. Also make sure that all images in the file have alternative text. See *Optimizing Adobe PDF Files for Accessibility*, in the *Additional Reading* materials following this section.
- If you include Powerpoint Presentations on your site, you need to provide an accessible version of both text and graphics. For existing PPTs, see the *Powerpoint Accessibility Techniques* in our *Additional Reading* section. If you are creating new PPTs for use on the web, see Microsoft’s *Creating Accessible Presentations* listed below.

- If you have applets or multimedia objects on your site, provide a description of the function of the object or applet between the opening and closing **<object>** element for browsers that do not support this tag. For those that do, provide a description in the **title** attribute. Other requirements for multimedia are detailed in the next section.

```
<object name="MediaPlayer" id="MediaPlayer" width="213" height="45"
classid="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95" title="Video of
product demonstration."
…>
Video of product demonstration.
</object>
```

2.1.1 Examples

Wrong: **** (No alt attribute at all.)

Wrong: **** (Has an alt attribute, but no appropriate alternative text for a content-bearing image.)

Wrong: **<img src="learn.gif" alt="Learn More">** (The alternative text is generic; it provides no information about the destination of the link. And no punctuation.)

Right: **** (Appropriate alt text, punctuated correctly, including a title tag.)

Right: **<img src="map.gif" alt="Map showing business locations in New Jersey." longdesc="locations.html">**

Wrong:

****

****

(Spacer images and decorative images do not need descriptive alternative text.)

Right:

****

****

NOTE:
There should be no spaces between the quotes for non-content images.

2.1.2 Additional Reading

Creating Accessible Images
http://www.webaim.org/techniques/images/

Text Alternatives
http://www.jimthatcher.com/webcourse2.htm

Visual Disabilities
http://www.webaim.org/techniques/visual/

Optimizing Adobe PDF Files for Accessibility
http://www.adobe.com/products/acrobat/pdfs/pdfaccess.pdf

Powerpoint Accessibility Techniques
http://www.webaim.org/techniques/powerpoint/

Creating Accessible Presentations (Microsoft)
http://office.microsoft.com/en-us/assistance/HA011667681033.aspx

2.2 Providing Alternatives to Multimedia

(b) Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation.

Since multimedia programs are inherently auditory and visual, people with sight or hearing impairments cannot completely enjoy them. The purpose of this 508 requirement is to address this issue by providing equivalent alternatives for both groups.

- If you have video files in your web site, make sure that the audio portion of your video is captioned *and synchronized* so that it can be used by the hearing impaired. A transcript alone is not sufficient. Synchronized captioning is required so that someone reading the captions could also watch the speaker and associate relevant body language with the speech. Provide an audio description of the visuals for the sight-impaired.
- If you have audio-only information in your web site, provide a text transcript of the audio for the hearing impaired.
- If you have Flash programs, you must be aware of many potential accessibility issues:
  - alternative text for images
  - keyboard access for controls
  - hidden content
  - flicker
  - alternative sound

  Consult the first two items in the list below for specific recommendations and techniques for making Flash content accessible.

2.2.1 Additional Reading

Best Practices for Accessible Flash Design
http://www.macromedia.com/resources/accessibility/

Creating Accessible Macromedia Flash Content
http://www.webaim.org/techniques/flash/

Audio & Multimedia
http://www.jimthatcher.com/webcourse6.htm

Captions
http://www.webaim.org/techniques/captions/

Hearing Disabilities
http://www.webaim.org/techniques/hearing/

2.3 Providing Alternatives to Color Indicators

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

In the United States alone, approximately 15 million people suffer from blindess or some other visual impairment. Approximately 8 – 10% of males and .5% of females suffer from some sort of color blindess. The purpose of this requirement is to ensure that people who fall into one of these two categories are not shut out from distinctions in content created through visual cues.

- Make sure that all information conveyed with color is also expressed without color, for example, from context or markup. Required fields, cautions or warnings, color-coded maps and diagrams, and other material specifically called out should not be indicated by a difference in color alone, but by asterisks, explanations, text labels, internal markup or alternative presentations. Examples include:
  - icons of different shapes (with alternative text) for color-coded items
  - user selectable paths for presentation of information that would otherwise be color coded, e.g. link to all sites with outages rather than color coding outages
  - text and table-based alternatives for map-intensive information
- Provide clear distinctions (sufficient contrast) between links and standard text and between visited and non-visited links.

2.3.1 Examples

Table 1. Example of non-compliant visual cue.

Required fields are marked in red

Last Name:		First Name:	
Tel #:		Email:	

Figure 1. Example of compliant visual clue.

2.3.2 Additional Reading

Creating Accessible Web Sites: Color Schemes and Backgrounds
`http://tlt.its.psu.edu/suggestions/accessibility/color.html`

Special Cases
`http://www.jimthatcher.com/webcourse7.htm`

2.4 Providing Alternatives to Stylesheets

(d) Documents shall be organized so they are readable without requiring an associated style sheet.

The primary purpose of this requirement is to ensure that users with screen readers are able to understand the structure and content of your web page.

Cascading Style Sheets (CSS) allow the presentation of content to be separated from the content itself. A different style sheet can be used to generate a look and feel optimized for web display, print, or a handheld device. For users with poor eye-sight, this can be a real boon, because they can apply a user-defined style sheet to your content to increase the point size.

CSS can also can be a liability, however, since the position of the content in your HTML page has nothing to do with its presentation. Banner information can potentially appear after footer information in the HTML page, for instance. The CSS will put them where they belong when presented through the browser to a sighted user. This is a problem for screen readers however.

When blind people browse your site with a screen reader, what they hear is the *linear reading order* of the content in your code. If the code contains the footer information before the banner information, the footer will be read first. You must keep this in mind when coding your HTML.

Another feature of CSS allows you to create, in effect, new elements using **`divs`** and **`classes`**. This makes it wonderfully flexible. But this too is a potential accessibility problem. Screen readers recognize and differentiate HTML structural elements. A first level heading is announced as such, and it is possible to move from heading to heading, enabling the user of the screen reader to “skim” the page to understand its structure. If you have used CSS to create visual equivalents of structural elements, however, screen readers will have no way of conveying that to a user.

- Make sure that the sequence of information/content in your HTML page is appropriate; it still makes sense when the style sheet is turned off.

NOTE:

Assuming you have no previous user-defined styles already, you can test this by opening your page with the Opera browser and selecting the User mode.

- Make sure that none of the information needed to use your pages is lost when the style sheet is turned off.
- Never use style sheets to create structures for which there is already an HTML element—such as headings, lists, and paragraphs. Instead, use CSS to style those elements appropriately.
- Make sure that there is nothing in your stylesheet to prevent users from calling in their own style sheet. Some users, for instance, may employ a user style sheet to raise the font size and adjust colors.
- Avoid inline styles, since they may not be able to be overwritten by an alternative style sheet.

2.4.1 Additional Reading

Creating Accessible Cascading Style Sheets
`http://www.webaim.org/techniques/css/`

Cascading Style Sheets
`http://www.jimthatcher.com/webcourseb.htm`

2.5 Providing Alternatives to Image Maps

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

(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.

Assistive technologies, like screen readers, need something to grab onto in the code presented to the browser. It can be explicit, such as a link or piece of text on the page, or implicit, such as a piece of alternative text.

From a screen readers point of view, a server-side image map presents an image, a set of “hot” coordinates, and a request to the server to establish the link. There is no URL or description of the link. This makes it inaccessible unless an alternative is supplied.

A client-side image map is preferable because there is always an explicit URL associated with the image coordinates. But unless there is a description of the destination, the user is left to try the link to find out where it goes.

This 508 requirement attempts to address both accessibility issues.

- Avoid image maps when possible. Most graphics can be "sliced" so that each slice has only one hot reference.
- If you must use an image map, use a client-side image map with appropriate **`alt`** and **`title`** attributes for each **`<area>`** element so that screen readers can announce the necessary descriptive link information to sight-impaired users.
- If you must use a *server-side* image map, include explicit text links elsewhere on the page for all items contained within the image map.

2.5.1 Examples

Wrong: (No alternate text on map items.)

```
<map name="navigation">
<area shape="rect" coords="0, 0, 200, 20" href="services.html" />
<area shape="rect" coords="0, 20, 200, 40" href="contacts.html" />
</map>
```

Right: (Has both alt and title attributes.)

```
<map name="navigation">
<area shape="rect" coords="0, 0, 200, 20" href="services.html"
 alt="Our Services." title="Our Services." />
<area shape="rect" coords="0, 20, 200, 40" href="contacts.html"
 alt="Contact Us." title="Contact Us." />
</map>
```

2.5.2 Additional Reading

Image Maps
`http://www.jimthatcher.com/webcourse5.htm`

Creating Accessible Images
`http://www.webaim.org/techniques/images/5`

2.6 Making Data Tables Accessible

(g) Row and column headers shall be identified for data tables.

(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.

The biggest problem for a person using a screen reader to access the web is navigating through tables. Part of the problem is that HTML pages often use tables for layout as well as for structuring data, so there needs to be some way of distinguishing data tables from format tables. The other major problem is envisioning/hearing the relationship between data cells and their associated data heads. Section 508 attempts to address both these problems by insisting that all data

tables contain explicit markup to associate cells and heads. Format tables should *not* contain such markup.

In addition to these minimal requirements, however, there are additional best practices that we strongly recommend (such as the use of caption and summary) to enable users with screen readers to decide quickly and easily whether they can skip a table. See *Other Accessibility Guidelines* for specifics.

- Whenever possible, design data tables as simply as possible, using only one level of column or row headers.
- Since screen readers read from left to right and from top to bottom, design data tables that read logically.
- Use the **<th>** element for the row and column headings of all data tables.
- Use the **scope** attribute to indicate whether a header is a row header or a column header.
- Use a combination of the **<th>** element and the **scope**, **id**, and **header** attributes to mark up complex tables.

2.6.1 Examples: Simple Table

Wrong: (Table has no caption, summary, or headers.)

```
<p><b>Simple Table:</b></p>
<table>
<tr>
<td>Column 1</td>
<td>Column 2</td>
<td>Column 3</td>
</tr>

<tr>
<td>Data A</td>
<td>Data B</td>
<td>Data C</td>
</tr>

<tr>
<td>Data A2</td>
<td>Data B2</td>
<td>Data C2</td>
</tr>
</table>
```

Right: (Uses headers and scope to define structure, and provides summary and caption information to enable a screen-reading users to decide whether they need to review this table.)

```
<table summary="A simple table provided as illustration of correct
coding principles.">
<caption>Simple Table</caption>
<tr>
```

```
<th scope="col">Column 1</th>
<th scope="col">Column 2</th>
<th scope="col">Column 3</th>
</tr>

<tr>
<td>Data A</td>
<td>Data B</td>
<td>Data C</td>
</tr>

<tr>
<td>Data A2</td>
<td>Data B2</td>
<td>Data C2</td>
</tr>
</table>
```

2.6.2 Examples: Complex Tables

Right*:* (Uses caption and summary to alert users to the contents of the table. Uses headers, scope, and ids to denote relationships.)

```
<table summary="A complex table illustrating the use of headers and
identifiers.">
<caption><strong>Table Title</strong></caption>
<thead>
<tr>
<th id="c0" scope="col"></th>
<th id="c1" scope="col">Column 1</th>
<th id="c2" scope="col">Column 2</th>
<th id="c3" scope="col">Column 3</th>
<th id="c4" scope="col">Column 4</th>
<th id="c5" scope="col">Column 5</th>
</tr>
</thead>

<tr>
<th id="r1" scope="row">Row 1 Header</th>
<td headers="r1 c1">Data</td>
<td headers="r1 c2">Data</td>
<td headers="r1 c3">Data</td>
<td headers="r1 c4">Data</td>
<td headers="r1 c5">Data</td>
</tr>
<tr>
<th id="r2" scope="row">Row 2 Header</th>
<td headers="r2 c1">Data</td>
<td headers="r2 c2">Data</td>
<td headers="r2 c3">Data</td>
<td headers="r2 c4">Data</td>
<td headers="r2 c5">Data</td>
</tr>
</table>
```

2.6.3 Additional Reading

Creating Accessible Tables
http://www.webaim.org/techniques/tables/

Techniques for Accessible Tables
http://www.ferg.org/section508/accessible_tables.html

Bring on the tables
http://www.456bereastreet.com/archive/200410/bring_on_the_tables/

2.7 Making Frames Accessible

(i) Frames shall be titled with text that facilitates frame identification and navigation.

The purpose of this requirement is to enable users with screen readers or other assistive technologies to quickly identify and locate frameset components. Since screen readers do not take a consistent approach to frame support, you need to be particularly thorough.

- Provide a one word description of the purpose of the frame in the name attribute of the **<frame>** element, for example, **<frame name="navigation1">**.
- Provide a more detailed description of the frame in the **title** attribute of the **<frame>** element, for example, **<frame title="Main Navigation">.**
- Provide an identical or consistent description of that frame in the HTML **<title>** element for the frame, for example, **<title>Main Navigation</title>.**
- Include a **<noframes>** option for those users who either cannot, or choose not to, view frames. On the **<noframes>** page provide a description of the frames and links to the individual frame pages as appropriate.
- Be sure to use the appropriate DTD for a frameset, for instance:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

- For I-frames, provide a one word description of the **<iframe>** element in the **name** attribute for scripting. To make the frame accessible to screen readers, use the **title** and **longdesc** attributes for brief and long descriptions of the frame respectively. For browsers that do not support I-frames, include either a copy of the content in the I-frame or a link to that content between the opening and closing tags.

2.7.1 Examples

Wrong*:* (Frames are not named. No alternate content for users without a frames-capable browser.)

```
<frameset cols="160, *">
<frame src="navigation.html" />
<frame src="content.html" />
<noframes>
Your browser does not support frames.
</noframes>
</frameset>
```

Right: (Frames are named appropriately. Alternate content provided for users without frames-capable browsers. In addition, frames are set in percentages making it easier for viewers with different screen sizes to access the material.)

```
<frameset cols="25%, *">
<frame name="navigation" src="navigation.html" title=”Navigation.”/>
<frame name="content" src="content.html"  title=“Primary content.”/>
<noframes>
Your browser does not support frames. Please visit the
<a href="noframes.html">no frames</a> version.
</noframes>
</frameset>
```

Right: (I-Frame contains appropriate descriptive text using the title attribute and alternative text for browsers that do not support I-frames.)

```
<iframe src=”ad.html” title=”CallVantage ad.”>
<p>Your browser does not support I-frames.</p>
<p>See our <a href=”ad.html”>CallVantage ad.</a></p>
</iframe
```

2.7.2 Additional Reading

Creating Accessible Frames
http://www.webaim.org/techniques/frames/

Making Accessible Frames
http://www.doit.wisc.edu/accessibility/online-course/standards/frames.htm

Using inline frames
http://www.cs.tut.fi/~jkorpela/html/iframe.html

2.8 Eliminating Flicker

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

For some people, intense blinking or flickering images, particularly in the frequency range of 2 Hz and 55 Hz can cause epileptic seizures. This requirement is intended to ensure that web sites do not unintentionally create this result.

NOTE:
The frequency of a typical Windows cursor is approximately .8 Hz.

- In general, avoid creating content that blinks, flickers, or alternates in quick succession.
- If you must have content which flickers in any way (e.g. animated gifs, Flash movies, etc.) make sure you provide a way for the user to stop the action.

2.9 Providing a Text-Only Page

(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.

In today's world, this alternative should almost never be necessary. Changes in HTML and advances in assistive technology together make it more appropriate and economical to make your primary site accessible, rather than trying to maintain a second, text-only version of the site.

- Before creating a text-only version of a web-page, make sure that there is truly no other alternative following other guidelines in this document.
- If you create a text-only page, you must ensure that it is equivalent in both function and information, *and* that it is kept current.

2.10 Making Scripts Accessible

(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.

The primary purpose of this requirement is to ensure that users who have motor disabilities and cannot use a mouse, or users with vision disabilities using a screen reader, can access the information associated with the script and launch it with a keyboard or other assistive device.

The 508 requirement is not as restrictive as the corresponding guideline in the World-Wide Web Consortium Accessibility Guidelines, which require your web page to function if scripting is turned off. But as a result, it is also less clear cut. Whenever possible then, adopt a policy of making sure that your page will work if scripting is turned off by the user. If that's not possible, follow the guidelines below, but be aware that your page should be tested with a screen reader or other assistive technology to make sure it passes.

- Make sure that buttons, links, and other objects that launch a piece of JavaScript code have **`alt`** text or **`title`** attributes that enable a screen reader to understand its function.
- Never use device-dependent JavaScript event handlers as *the only way* to display or select critical information. Device-dependent event handlers include
  - **`onmouseover`**
  - **`onmouseout`**
  - **`ondblclick`**
- If you must use such events handlers, provide redundant device-independent event handlers to allow keyboard access or alternative ways of displaying information.
- Always ensure that JavaScript content such as menus, submenus, selectable options are accessible via keyboard and assistive technologies such as screen readers.
- Avoid automatic updating of content in the browser without an explicit command from the user. Or, at least provide a mechanism for explicitly requesting an update.

NOTE:

Providing content via a **<noscript>** element is a necessary alternative for users whose browsers do not support scripting or who simply wish to turn off scripting for security reasons. But it is *not* the complete solution for 508 compliance. Since most screen readers enable JavaScript by default, the **<noscript>** page will rarely, if ever, be discovered. A script must be accessible to assistive technologies.

2.10.1 Additional Reading

Creating Accessible JavaScript
http://www.webaim.org/techniques/javascript/

Scripts and Applets
http://www.jimthatcher.com/webcoursea.htm

2.11 Providing Links to Compliant Applet/Plugin Pages

(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).

The purpose of this requirement is twofold: to ensure that users wishing to use a web-based application can easily install any additional software required, and to reinforce the message that both the primary code for an application and any additional code from a plugin must be 508 compliant.

- Provide a reliable link to any plugin, such as RealPlayer or Adobe Acrobat that is required by your application.
- Ensure that any plugin you use meets 508 requirements.

2.12 Making Forms Accessible

(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.

The primary purpose of this requirement is to allow users with screen readers to easily access, understand, and complete web-based forms. There are two aspects to this requirement. First, since a blind user cannot use a mouse to make selections, all controls and selections have to be accessible and selectable from the keyboard. Second, since screen readers read in sequence from left to right and top to bottom, fields need to be organized in an appropriate sequential order *or*, even better, additional code needs to be added so that the fields and their labels are always properly associated.

- Never base the selection of a form control upon a JavaScript event that is solely mouse-dependent. For instance, never associate the selection of a specific option from a pull down list with **onmouseover** or **onmouseout**.

- Whenever possible, design forms so that they linearize well, that is, that form fields and their labels are properly associated when a screen reader reads through your HTML page in left to right and top to bottom fashion.
- As a precaution, always use the **<label>** element to specifically associate field names with their related input fields.

```
<label for="name_last" title="Last Name.">
<input id="name_last" type=”text” size="20" maxlength="120"  />
<span class="required">Last Name *</span>
</label>
```

- Because it is easy for screen readers to lose a sense of context when there are multiple options such as radio buttons or checkboxes associated with a given choice, be sure to use the HTML **<fieldset>** and **<legend>** elements to group form controls.

```
<form action="http://www.icecream test.att.com">
<fieldset id="favorite_ice-cream">
<legend>Choose Your Favorite Ice Cream</legend>
<input id="vanilla" type="radio" name="radio" value="vanilla">
<label for="vanilla" title="Vanilla">Vanilla</label>
<input id="chocolate" type="radio" name="radio" value="chocolate">
<label for="chocolate" title="Chocolate">Chocolate</label>
<input id="strawberry" type="radio" name="radio" value="strawberry">
<label for="strawberry" title="Strawberry">Strawberry</label>
<input id="orange" type="radio" name="radio" value="orange">
<label for="orange" title="Orange">Orange Sherbet</label>
</fieldset>
</form>
```

- Use the tabindex attribute to enable users to tab to each field in logical succession rather than use a mouse.
  <input id=”name_last” type=”text” tabindex=”1” />
- Provide a descriptive **<label>** element for every **<select>** element.

```
<label for=”states”>State</label>
<select id=”states”>
<option value=”alabama”>Alabama</option>
option value=”arkansas”>Arkansas</option
</select>
```

- When it makes sense, use <optgroup> elements to organize options.

```
<select title="List of Fruit trees" name="fruit-trees" id="fruit-
trees">
   <optgroup label="Citrus">
   <option value="Orange">Orange</option>
   <option value="Lime">Lime</option>
   <option value="Lemon">Lemon</option>
   </optgroup>
   <optgroup label="Deciduous">
   <option value="Apple">Apple</option>
   <option value="Pear">Pear</option>
   <option value="Peach">Peach</option>
```

```
</optgroup>
</select>
```

- If you use a graphic rather than a standard form button, be sure to include the appropriate alternative text.

  <input type="image" alt="Submit" src="/images/submit.gif" />

2.12.1 Examples

Right:

```
<form action="http://www.formtest.att.com">
<fieldset id="customer_info">
<legend>Customer Information</legend>
<label for="name_last" title="Last Name.">
<span class="required">Last Name *</span>
<input size="20" maxlength="120" id="name_last" /></label>
<label for="name_first" title="First Name.">
<span class="required">First Name *</span>
<input size="20" maxlength="120" id="name_first" /></label>
</fieldset>
</form>
```

2.12.2 Additional Reading

Accessible Forms
http://www.jimthatcher.com/webcourse8.htm

Creating Accessible Forms
http://www.webaim.org/techniques/forms/

2.13 Enabling Users to Skip Links

(o) A method shall be provided that permits users to skip repetitive navigation links.

Sight-impaired users don't want to hear the same set of navigation items read over and over on every page they visit. They want an easy way to skip over links to get to the primary content of the page.

- Provide skip links on every page just after the **<body>** element. You can hide the skiplinks from sighted users by camouflaging the links using a CSS rule that defines the skiplink to be a very small point size and the same color as its background location. This technique makes the link "invisible" to sighted users, but includes it in the code for the page so that screen readers can read it.

2.13.1 Example

```
<!-- 508 Compliant Skip Navigation Link -->
<a href="#grid" title="Skip Navigation Links." id="skip">Skip
Navigation Links</a>
<!-- /508 Compliant Skip Navigation Link -->

CSS
/*---------------508 Compliant Skip Navigation Link-----------------*/
#skip{
```

```
display:inline;
color:#FFF !important;
text-decoration:none;
font-size:1px;
position:absolute;
top:0;
left:0;
z-index:10;
}
```

NOTE:

This solution and the associated CSS for it are already included in the latest version of the Reusable Asset Library Templates.

2.13.2 Additional Reading

Skip Navigation Links, **`http://www.jimthatcher.com/skipnav.htm`**

2.14 Facilitating Timed Reponses

(p) When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required.

People using assistive technologies, such as a screen reader, may take longer to understand a page, fill out a form, or respond to a question. This requirement enables them in effect, to ask for more time.

- Provide an accessible warning or message when a timed response or action is required.
- Enable the user an easy method to extend the time.

3. Other Accessibility Guidelines

- Keep **`alt`** text brief, but descriptive. Remember a sight-impaired user wants to be able to skim a page for information just like you do.
- If you have “Learn More” or links and images without context, makes sure you provide the context for that link in the alternative text.
- Avoid fixed fonts. Sight-impaired users may want to adjust the font size up or down.
- Use the **`<strong>`** and **`<em>`** (emphasis) tags in place of **`<b>`** or **`<i>`**. There is no difference in the way a text reader pronounces italic or normal text. When text is placed within the **`<em>`** tag, however, that text is emphasized.
- Use the **`<caption>`** element to provide a short description of a data table. Be sure to place the **`<caption>`** element immediately after the **`<table>`** element. This helps a screen reader immediately identify the usefulness of a table.
- If the table is complex or not adequately described by the caption, use the **`summary`** attribute of the **`<table>`** element to provide a more detailed description of the table structure and contents to help orient the user to what’s coming.

- Use **<noframes>** element to make content accessible to devices that don't support frames
- Do not create **divs** or **classes** for functions or structures for which there is already an HTML element.

4. Accessibility Standards & Resources

Although accessibility for people with disabilities is the primary focus of this document, creating accessible web content means more than that. It also means making content easy to find in a search engine, accessible to browsers other than Internet Explorer, and potentially accessible to a range of other user agents.

The references below reflect the view that accessibility is not a burden but an opportunity to make AT&T content available to a much broader base of users, thereby increasing the bottom line.

4.1 Standards

Section 508 Law
http://www.section508.gov/index.cfm?FuseAction=Content&ID=3

Summary of Section 508 Standards
http://www.section508.gov/index.cfm?FuseAction=Content&ID=11

Web Content Accessibility Guidelines 2.0
http://www.w3.org/TR/WCAG20/

4.2 Resources

4.2.1 Government

Federal Agency Public URLs on Section 508 Guidance
http://www.section508.gov/index.cfm?FuseAction=Content&ID=131

Guide to the Section 508 Standards for Electronic and information Technology
http://www.access-board.gov/sec508/guide/index.htm

4.2.2 Other

508 Web Accessibility
http://www.jimthatcher.com/webcourse1.htm

Accessify Forum
http://www.accessifyforum.com/

A List Apart: Accessibility
http://www.alistapart.com/topics/accessibility/

Constructing Accessible Web Sites, Jim Thatcher, Glasshaus Ltd.:Birmingham, UK, 2002.

CSS Techniques for Web Content Accessibility Guidelines 1.0
http://www.w3.org/TR/2000/NOTE-WCAG10-CSS-TECHS-20001106/

CSS Techniques for WCAG 2.0
http://www.w3.org/TR/2004/WD-WCAG20-CSS-TECHS-20041119/

How People with Disabilities Use the Web
http://www.w3.org/WAI/EO/Drafts/PWD-Use-Web/

Observing Users with work with Screen Readers
http://portal.acm.org/ft_gateway.cfm?id=947227&type=html

Web Accessibility in Mind
http://www.webaim.org/techniques/