

1. Introduction

Until recently, most government Web sites were constructed much like any commercial Web site. They used a table-based layout with tables nested within the layout table, Javascript effects, tags that had been around since HTML 2.0, and more attention to the “gee-whiz” factor than anything else.

Since there were many differences between browsers, some developers created a couple of different versions of the Web site and switched users to a certain version depending on what type of browser was used. All this made for very large pages and multiple opportunities for things to go wrong.

As Web browsers have matured to comply with World Wide Web Consortium (W3C) standards, however, the time is right for Web developers to begin writing to those standards. This will ultimately make Web development and maintenance easier and help Missouri government Web developers to reach all their potential users.

Missouri government Web sites have certain standards that they must follow. The main standard is accessibility—making sure that all constituents can read the content of the site, regardless of physical abilities or browser preference. However, another goal has to be usability—how easy will it be for a constituent to find the information she is looking for? A third objective is integration with the state portal—can the user tell that the site is an official Missouri government site? Does the page link back to the state home page?

The DMD believes that following the W3C Web Standards, Missouri government Web developers will meet most of the accessibility standards without having to resort to expensive third-party tools to create accessible versions of their Web sites.

These standards and guidelines have been used by many Missouri government agencies and serve as a “best practices” guide for other state agencies to develop effective Web sites for the Missouri State Web.

Using these guidelines will help agencies:

- < Develop accessible Web pages
- < Make more usable sites
- < Communicate more effectively
- < Make maintenance easier
- < To communicate information efficiently and effectively
- < To improve the ease of use and organization of information on state Web sites
- < To enhance the interoperability of state Web servers and Web sites
- < To meet user expectation of information and technology

A. Scope

The guidelines are concerned with the Web developer’s final product, i.e., what is rendered on the browser, as opposed to any particular server technology used to produce the final code. Thus, we will not address techniques specific to ASP, .NET, JSP, PHP,

CFM, Perl or any other scripting languages.

The guidelines are not limited to just the Internet, though. They apply to all browser-enabled applications, whether on the Internet, an intranet or an extranet. Following the guidelines will help to make sure all these applications meet Missouri standards.

The items presented here tend to fall into one or more of three categories:

1. Common elements
2. Usability
3. Accessibility

Common elements are those items that we recommend that all state Web pages have. Examples include navigation tools, links, meta data and icons.

Usability items help users use the site more effectively or help the Web developers maintain the site more easily.

Accessibility standards are items that address the OIT accessibility Standards, [http://oit.mo.gov/standards/ITGS0003 Missouri IT Accessibility Standards.doc](http://oit.mo.gov/standards/ITGS0003_Missouri_IT_Accessibility_Standards.doc)

B. Authors

This document was drafted by members of the Digital Media Developers (DMD) group, a subcommittee of ITAB composed of state government employees involved in issues relating to Web and digital media development. Principal contributors to this revision were Debbie Boeckman (DNR), Connie Farris (MSHP), Erica Gage (SOS), Debbie Karaff (DSS) and Kevin Lanahan (MDI).

You can find out more about the DMD at <http://www.oa.mo.gov/dmd/>.

2. Standards

Missouri government Web sites should support the Missouri [IT Accessibility Standard](#), the current Web standard, [XHTML 1.0](#) and Cascading Style Sheets ([CSS2](#)). This will involve using techniques in any redesign or redeployment that may be new to some Web developers.

Web standards have finally matured. Web Accessibility guidelines have been standardized since 1999, federal law Section 508 has been in effect since 1998, XHTML 1.0 has been a W3C recommended standard since 2000 and XHTML 1.1 (modular XHTML) has been recommended since 2001. CSS1 has been recommended since 1996 and CSS2 since 1998.

If you write your code to standards, your pages will be mostly accessible by default.

This pause in Web standards evolution has allowed browsers to implement most of the features of these standards. This means that you, the Web developer, can expect a standards-compliant browser to behave in a predictable manner. That means you no longer have to do a lot of work-arounds or browser detection to make sure your sites are viewable in most browsers.

For accessibility, this means that you don't have to use third-party applications to create "text-only" pages for accessibility. By using standards, your Web pages are accessible by default.

XHTML (and HTML 4) and CSS allow for separation of content and presentation, which means that any page can be structured very simply, using semantic markup and no presentational markup. All the presentation information is contained in the style sheet, which instructs the browser how to display the tags in the Web page.

What this means is that you can create a very simple text-only page and mark it up to display any way you please; users that can't use style sheets will only see the plain text page and users with current browsers will see the pretty page. This is a great step towards accessibility.

Note: Each state agency has the freedom to design and develop a Web site appropriate to inform and serve their audience. Different agencies have different resources and skills, and they should be free to use them to their best abilities.

However, there are some best practices which can make the development and maintenance of a site easier. In addition, there are common elements that will help brand each Web site as being part of the State of Missouri Web and make navigation between different agencies easier for the public.

These guidelines place no restrictions on the design and development of a Web site; they are recommendations to ensure a successful deployment of a Web site.

A. Moving to Standards

Moving to a standards-based format will mean you will have to learn some new things and relearn some old things.

XHTML has a few differences from HTML that are easily learned and a few differences that will have to become habits over time.

Cascading Style Sheets have been around for a few years, but browser support for CSS2 is only a few years old, and you will probably want to get a good book to help explain all CSS can do for you.

B. Accessibility

Accessibility is not optional; it is mandated by state statute [Section 191.863 RSMo](#). Missouri Web sites must follow the Missouri IT Accessibility Standard (<http://oit.mo.gov/initiatives/itaccessibility.html>), which follows the [Federal 508 Web accessibility standards](#) with just a couple of exceptions.

The need to make Missouri Web sites accessible drives the need to follow standards in Web design. A well-designed standards-compliant site will be mostly accessible because it will be a basic, well-formed text page with some presentation styling.

C. XHTML Rules

There are ten major requirements for XHTML that are not in HTML.

- 1) You must use a proper DTD.
- 2) You must use a proper namespace.
- 3) You must declare your content type.
- 4) You must make all your tags lowercase.
- 5) You must quote all attribute values.
- 6) All attributes require values.
- 7) You must close all tags.
- 8) Even empty tags, like
.
- 9) You can't use double dashes in comments, and
- 10) You need to encode all < and & characters.

1) Document Type Definition (DTD)

Each XHTML page should have a Document Type Definition (DTD or Doctype) as the first line of the page.

The DTD tells the browser what set of tags to interpret in your page. XHTML specifies three DTDs, so authors must include one of the following document type declarations in their documents. The DTDs vary in the elements they support.

The [XHTML Strict DTD](#) includes all elements and attributes that have not been [deprecated](#) or do not appear in frameset documents. For documents that use this DTD, use this document type declaration:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

The [XHTML Transitional DTD](#) includes everything in the strict DTD plus deprecated elements and attributes (most of which concern visual presentation). For documents that use this DTD, use this document type declaration:

```
<!DOCTYPE HTML "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

The [XHTML Frameset DTD](#) includes everything in the transitional DTD plus frames as well. For documents that use this DTD, use this document type declaration:

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

If you are using HTML, DTDs can be found at <http://www.w3.org/TR/html401/sgml/dtd.html>.

The DTDs tell the browser how to interpret the code in your page. XHTML-Transitional is closest to regular HTML, and is most forgiving of invalid code.

Most Web editors will either insert a DTD automatically or can be configured so the main template will automatically insert one.

Strict or Transitional?

Transitional (also known as “quirks”) mode kicks in whenever a DTD is not declared, or if your document does not validate. It is the default DTD for Internet Explorer 6. Writing to a transitional standard allows you to use some deprecated attributes, like “*bgcolor*.”

Strict mode doesn’t cut you any slack. Your code must validate or your page will get displayed using quirks mode. This means you have to be a little more careful when creating your pages, but you will have greater control over your page. IE6 will display strict mode properly, so you will have a consistent look no matter which browser your users use.

So which one do you use? If you are uncomfortable with changing the way you have always created pages, Transitional mode may be the one for you. It will give you a little more time to clean up sloppy code and tighten up your pages. But remember, it is *transitional*. You really need to think about switching over to Strict sooner or later.

If you make the jump into Strict mode, you may be confused early on as you get used to the no-nonsense rules, but they soon become second nature. Some development tools, like Dreamweaver MX, will automatically convert your pages to comply with your DTD.

2) *Namespace*

The namespace is an extension of the basic <html> tag found at the top of each old Web page. It follows immediately after the DTD statement, and looks like this:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

The first part of this statement, *xmlns="http://www.w3.org/1999/xhtml"*, tells the browser where to find the namespace on the web.

The other two parts, *xml:lang="en" lang="en"*, indicate that the XML and page language is English.

3) *Content type*

The actual W3C XHTML specification calls for an <xml> statement above the DTD, where you declare the character encoding for the document. Unfortunately, if you do this, IE will automatically render the page in quirks mode, which can mess up your layout. Other browsers may actually crash.

Instead, you will insert a meta tag in the head of the document, like this:

```
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />
```

In this example, the document uses the ISO-8859-1 character set (also known as Latin-1). There are other character sets available if you are coding in other languages or need an internationalized character set.

4) *Lowercase tags*

XHTML is case-sensitive. All elements and attributes (tags and properties) must be in lowercase in order to validate. For example, this is invalid code:

```
<OL CLASS="bigfont"><LI OnMouseOver="doJS">Item 1</LI><LI>Item  
2</LI></OL>
```

but this is valid code:

```
<ol class="bigfont"><li onmouseover="doJS">Item 1</li><li>Item 2</li></ol>
```

5) *Quote attribute values*

This is pretty simple. Instead of having code like this:

```
<table cellpadding=0 cellspacing=2 border=1>
```

You will need to use quotes:

```
<table cellpadding="0" cellspacing="2" border="1">
```

You also need to separate each attribute with a space.

6) *All attributes require values*

Any attribute that was just a word in HTML will need to have a value declared as well.

- HTML: <input type="radio" name="radio1" value="5" selected>
- XHTML: <input type="radio" name="radio1" value="5" selected="selected">

7) *Close all tags*

HTML let you assume a tag closed when you started a new one, like this:

<p>The quick red fox jumped over the lazy brown dog.
<p>She sells seashells by the seashore.

XHTML makes you close those tags, which is a good habit to get into anyway:

<p>The quick red fox jumped over the lazy brown dog.</p>
<p>She sells seashells by the seashore.</p>

8) Close the empty tags, too

Tags like
, <hr> and do not have a corresponding closing tag (like </br>, </hr> or).

In XHTML, you close the tag by adding a space and a forward slash before the closing >, like this:
, <hr /> and

9) No double dashes in comments

XHTML only allows a double dash (“—”) at the beginning and the end of a comment. So this is not allowed:

<!--This is a bad comment—no fooling. -->

But you can put a space between the dashes, so this is valid:

<!--This is a valid comment- -for real. -->

10) Encode special characters

If you use an ampersand, greater than sign, less than sign or quote (“&”, “>”, “<” and “””, respectively) in your text, you will need to encode them like this: &, >, < and ".

Those characters are reserved in XML, but XHTML validators will let you off with a warning.

D. As you create new pages...

Keep the following in mind when creating your pages:

1. Use valid HTML.
2. Content should be separate from style.
3. Don’t make pages too long so that there is endless scrolling, unless you provide “anchors” and links to aid the reader in navigation.
4. Avoid using deprecated tags.

The following tags have been deprecated and should no longer be used. HTML 4.01 will allow you to use them, but XHTML 1.0 forbids them:

< CENTER
< FONT
< U (underline)
< APPLET

These tags should be replaced with appropriate style sheets or standards-compliant tags.

E. Cascading Style Sheets

A major component of XHTML and HTML 4 standards is the use of Cascading Style Sheets (CSS). CSS allows the Web author to separate Web content from Web style. By creating a style sheet, you can greatly reduce the amount of in-line styling (FONT tags, tables and link colors, for example), which will, in turn, make updates and redesigns easier.

1) *Advantages of CSS*

The first big advantage of CSS is code reduction. All the CSS can be stored in an external style sheet that is stored in cache, so after it initially loads it is available for all subsequent pages. And all the table and presentational code that you currently use would disappear. You can expect a 10-50% reduction in code, depending on how clean your original code is.

Another advantage is ease of update. If you code all your <h1> tags to be 20 pixels high, and later decide to make them 25 pixels, you can change one line of code and the entire site is updated. Much, much easier than tickling through your entire site to change each and every tag by hand.

The third advantage is accessibility. You should be able to turn off style sheets and still read the entire page. That means any user with an obsolete browser or a screen reader should still be able to access all your content.

2) *Disadvantages of CSS*

Well, it is confusing at first. The terminology is different and it will take a while to learn what CSS elements and attributes correspond to the obsolete HTML attributes.

It doesn't work in older browsers. IE3, IE4 and Netscape 4x had incomplete CSS compatibility, and earlier browsers (and, of course, non-graphical browsers) do not have any support for CSS at all. So, if you dealing with a large population that only uses obsolete technology or does not use graphical browsers, they may not see your pages in all their styled glory. However, there is a good chance that they can't see your pages at all if you are using font tags and other in-line, non-semantic code.

A third disadvantage is that CSS can be addictive. Once you start working with it, you just can't stop.

F. Validation

Three words to ensure the success of moving to standards-based design:

Validate, validate, validate

In order to make sure your page displays the way your DTD says it should, you will have to validate it. Some Web development applications (like Dreamweaver) have validators built in, but there are a couple of on-line validators you can use as well.

- W3C HTML Validation Service: <http://validator.w3.org/>
- WDG Web Validator: <http://www.htmlhelp.com/tools/validator/>

Either of these tools will allow you to check a page on-line, and will also allow you to

check local pages before you upload them to your site.

After your HTML validates, you need to make sure your CSS validates. If you are using a third-party tool like TopStyle, you can set the level of compliance to hit individual browsers or code to CSS1 or CSS2. Any errors will be color coded.

Once you have your style sheet completed, you can test it against the W3C CSS Validation Service: <http://jigsaw.w3.org/css-validator/>

If your CSS validates, it's time for the last step. You need to check for 508 compliance with whatever accessibility tool is built into your development program, or use Cynthia Says or Bobby.

- Cynthia Says: <http://www.contentquality.com/>
- Bobby: <http://bobby.watchfire.com>

Both of these validators will identify any obvious accessibility problems, but will also give you some indications of what items must be checked manually.

There is another accessibility standard, the Web Access Initiative (WAI), which covers more issues but the issues are more difficult to test than the items in Section 508. While Missouri requires you to validate to Section 508, you may also want to test your pages against the WAI specifications. Both Cynthia Says and Bobby will test against either of the standards.

G. Browser Compatibility

In the days of the Browser Wars between Netscape and Internet Explorer, when new versions of browsers, each with its own unique feature set, would appear every few months, you had to be concerned with what browser to write to. Should you write for IE, since most of your users used it, and risk alienating your Netscape users (and your Mac users)? Or should you stick with HTML 3.2, since all the browsers knew what to do with pages written to that standard?

Well, the war may be over (IE won, for now), Microsoft hasn't updated their browser in a couple of years, the standards have stayed intact for several years, and other browsers, most notably those based on the Mozilla project, have worked very hard to implement those standards.

The result is that you don't have to talk about browser compatibility at all. All you need to be concerned about is accessibility. If you design your pages using standards and validate the pages before publishing, you will find that your pages will at least be readable on nearly any browser.

3. Common Elements

Common elements fall under two categories:

- < **mo.gov common elements:** elements that are common to all state web sites across all agencies, and
- < **agency common elements:** elements that are common to all the pages within an agency's site.

Common elements:

- < make your site a recognizable part of the Missouri Government web
- < give your users a sense of familiarity as they navigate through the State of Missouri web and your agency web site.
- < help standardize your web site and
- < help reinforce the need to use templates to develop your web pages.

mo.gov Common Elements

mo.gov common elements are items that should be on all state agency web sites. They fall into two major categories:

- < navigation elements and
- < design/layout elements.

A. Navigation elements

- *Link to State Home Page*

Standard: Each agency home page must have a link to the State of Missouri home page (<http://www.mo.gov>).

Guideline: The users should not have to scroll to find the link.

The link to the state home page should be easily identified in the main navigation of the agency home page, either with a graphic or prominent text.

A link to the State home page:

5. will allow users to navigate smoothly between the state portal and agency pages.
6. identifies your agency as part of the State of Missouri web.

B. Link to Search Page

Standard: Your home page should allow searching through the state search engine as well as a form to search the agency pages.

Guideline: Your search page should be customized to reflect your site design and default to search your site first.

A tutorial on customizing the search pages is available in the References section on the DMD web site (http://www.oa.mo.gov/dmd/links/Tips_and_Tricks/).

If you have your own search engine installed on your server you may use it, but you still must have a link to search using the state search engine. Many users rely on searching rather than navigation tools to find the information they need, so it is important to provide links to search tools that will let users find information in your site as well as related information in other agency sites.

C. Layout/Design elements

- 1) *DTD*

Standard: Every page must have a DTD declaration for each page.

Guideline: XHTML-Strict, XHTML-Transitional, HTML-Strict or HTML-Transitional

While the DTD is not exactly a layout or design element, it tells the user's browser how to render a page. Without this element, the browser will usually go into “quirks” mode and may not render your page as you expected.

Using a DTD will:

- < make your page load faster, by letting the browser know how to interpret your page,
- < give you more control over your presentation by specifying how the page should appear, and
- < help your site become more accessible by using valid code that assistive technology can interpret.

- 2) *Page Size*

Standard: Use a flexible design that will scale to the user's resolution and browser window.

Guideline: Test to a screen size of 800x600 pixels

A flexible design scales to the user's resolution, so it won't matter if a user has his monitor resolution set to 800 by 600, 1024 by 768 or 640 by 480, and it won't matter if he browses full-screen or in a window. You will want to use percentages instead of absolute widths for your page elements and design your graphics so they will scale with the size of the page.

Remember, too, that users may want to access your site on a PDA or even a cell phone. Those devices have much smaller resolutions.

Pages should be tested to a screen size of 800x600 pixels, which is the default screen resolution for many graphics cards and many users do not change from this size.

You should keep in mind that there are still a number of users who do use 640x480 resolution, either because they have an older system or because they have visual problems and prefer a lower resolution.

Make sure that pages are readable at lower resolutions. MSN TV (formerly Web TV) only has a 560 pixel width and no horizontal scrolling and cell phones may only have a 120 pixel width.

-

- 3) *Layout*

Standard: Your web site should have a consistent look and feel across all your pages.

Guideline: A simple layout consisting of a header, footer, sidebar and body is preferred. This style of layout is familiar to most users and is relatively simple to do without using any tables (Illustration 1).

An alternate layout may be to have both a left and right sidebar, or a right-side sidebar only.

Whenever possible, tables should not be used for layout. Your design will be much more flexible and easier to maintain if you use a table-less design.

The *header* should contain the agency logo (and other agency identification) and site-wide navigation tools (search, contact info, help). This header would appear on all pages in one version or another.

The *footer* is a good place to put your disclaimers and privacy statement.

This may also be the best place to put an accessibility link for any users that may have problems reading the page, as required by the Missouri IT Accessibility Standards.

The footer should contain a way for the user to contact the web master to report problems with the site, as well as the date the page was last updated and copyright information.

The *sidebar* is where you would put some topical navigation links. The sidebar should appear on most top-level pages on your site.

The *content* section contains the information or data that you are presenting to the user. This section will change on each page, while the other sections will be part of a template.

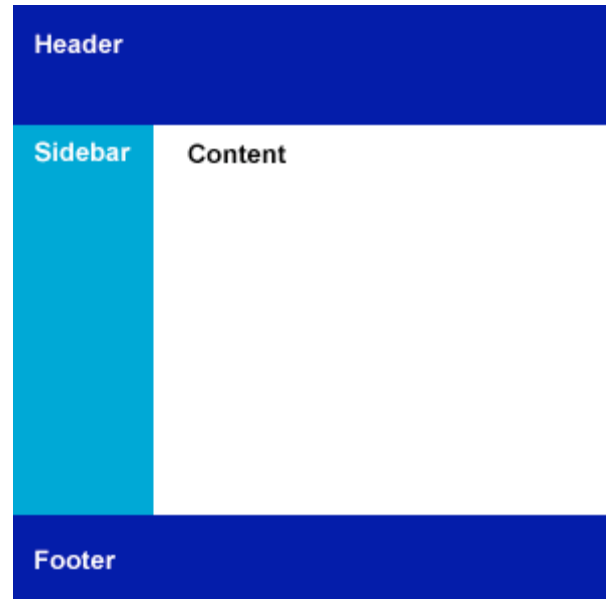


Illustration 1. Sample Layout

D. Agency Common Elements

These common elements should be on most pages throughout your web site. They will help brand your web site and reinforce your agency's web identity. All of these items are guidelines for the purpose of this document, but your agency can make them into an agency standard for your site.

- ***1) Navigation***

Guideline: You should also organize your site by functions rather than by the hierarchical organization of your agency.

Organizing this way will help the users find the services and information they need without knowing the political structure of the agency.

Guideline: If your agency is part of a department or other larger organization, the relationship to that organization should be made very clear, with links back to the parent organization in your main navigation.

Guideline: Your site should have a consistent navigation scheme across all pages.

Navigation should include a Search form and a contact link. It may also include such items as online services, employment links, FAQs or other pages that will help users navigate through the site.

Guideline: You should not use graphics (other than logos) for your navigation.

Most navigational links are text-based, and you can achieve most rollover effects using CSS rather than javascript and images. This will reduce your page size, and make your page more accessible and easier to maintain.

There is nothing wrong, however, with using background images or replacing bullets in a navigational list with images.

- ***2) Layout***

Guideline: Avoid using tables for laying out your site.

Instead, create your page as a plain text page, using hierarchical markup (<h1> for the main title of you page, <h2> for secondary topics, <p> for paragraphs, etc) and <div> tags to mark the different areas of the page.



[State Home Page](#) | [DMD Home Page](#) | [Search](#) | [Site Map](#)

- ◆ Item 1
- ◆ Item 2
- ◆ Item 3
- ◆ Item 4

Missouri Web Standards and Guidelines

Introduction

Scope

The guidelines are concerned with the web developer's final product..

This is the footer of the page.

Illustration 2--Basic HTML page with no styling

Look at the code: *<replace text with link to separate exhibit>*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
<title>Example 1</title>
<link href="/temp/DMD/css.css" rel="stylesheet"
type="text/css" />
</head>
<body>
<div id="header"> 
  <div id="global-nav">
    <p> <a href="http://www.mo.gov">State Home Page</a> | <a
href="/index.htm">DMD
      Home Page</a> | <a href="/search.htm">Search</a> |
<a href="/sitemap.htm">Site
      Map</a> </p>
    </div>
  <!--close top_navigation div -->
</div>
<!--close header div -->
<div id="sub-nav">
```



```

    <ul>
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
      <li>Item 4</li>
    </ul>
  </div>
<!-- close sidebar -->
<div id="content">
  <h1>Missouri Web Standards and Guidelines</h1>
  <h2>Introduction</h2>
  <h3>Scope</h3>
  <!-- Note the escaped apostrophe (&#39;) in the next line
-->
  <p>The guidelines are concerned with the web
developer&#39;s final product...</p>
</div>
<!-- close content section -->
<div id="footer">This is the footer of the page.</div>
<!-- close footer section -->
</body>
</html>

```

Then, use CSS to mark up the page, giving styles and positioning to the header, footer, sidebar and content sections.

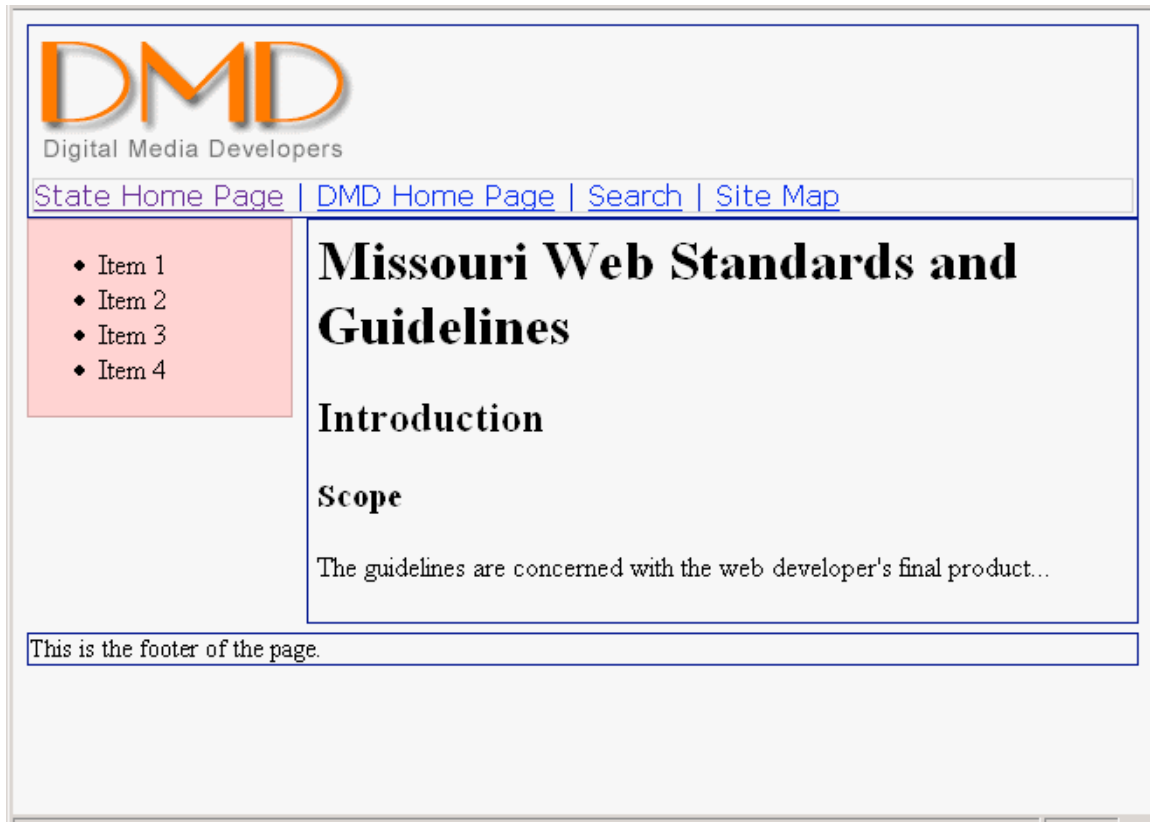


Illustration 3--Basic page with styles added

Look at the style sheet: <replace text with link to separate exhibit>

```
#header {
    background-color: #fff;
    color: #ccc;
    border: 1px solid navy;
    padding: 2px;}
#header img {border: 0;
}
#global-nav {
    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
    color: #00c;
    border: 1px solid #00f;
    border-color: #ccc;
}
#global-nav p { margin:0;}
#sub-nav {
    float: left;
    width: 150px;
    color: #000;
    background-color: #fcc;
    margin-right: 10px;
    border: 1px solid #c99;
```

```
}  
#content {  
    margin-left: 160px;  
    margin-bottom: 5px;  
    color: #000;  
    background-color: #fff;  
    border: 1px solid navy;  
    padding: 5px;  
}  
h1 {margin-top: 0;}  
#footer {  
    font-size: .9em;  
    border: 1px solid navy;  
}
```

•

- **3) A note about frames**

Frames are strongly discouraged on state Web sites. They have some serious accessibility issues, and make it difficult, if not impossible, to bookmark or link specific pages on your site.

Instead, consider using server-side includes to put persistent elements (like navigation sidebars, headers and footers) in your pages.

The following are tips to consider if you choose using frames:

- < Frames make it difficult for other sites to deep link to your site and maintain any navigation.
- < You will need to provide a “no frame” alternative to pages with frames.
- < Frames are not available on all browsers.
- < If overused, frames can create a page that is cluttered and difficult to use.

E. Page Elements

In addition to common elements, there are some basic design best practices for your site and for each page.

- **1) DTD**

Every page needs to have a DTD declared in the first line of the page, as specified in the Layout/Design Elements section.

- **2) Head content**

This is information that goes into your <head> tag on each page. It contains data that is used by the browser, search engines, and the user. This is an often-neglected section of the Web page. Here's what you need to include:

- **3) Title**

The title of the page is an important but often overlooked tag. The title gets picked up by most search engines and displayed in the search results, so a good title will help the user select the right page.

The title of pages should be short and descriptive of its contents. It is used by search engines and by the user's bookmarks. The title should indicate your state and agency wherever possible.

Example:

Bad title: Standards and Guidelines.

Good title: State of Missouri Web Standards and Guidelines.

- **4) Meta tags**

Meta tags contain information about the page. These can include author information, keywords or descriptions. Many search engines (including ht//dig, the state search

engine) use meta information to rank results.

Three meta tags are particularly important:

1. The “http-equiv” meta tag tells the browser what character set to use to display the page. It is vital to include this tag if you are using XHTML and you have omitted the `<?xml>` declaration at the beginning of the document.

The format for this meta tag is:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

2. The “keywords” and “description” meta tags help the state search engine, `ht://dig`, to index and rate your page. Other search engines may use these meta tags to rate search results, but most of them use the actual page content to rank your page.

Keywords are just words that would help identify the contents of the document or the purpose of the page.

`Ht://dig` looks for space-delimited keywords, like this:

```
<meta name="keywords" content="Missouri web standards DMD" />
```

Other search engines look for a comma-delimited list of words.

“Description” can be a phrase or paragraph that explains what the page contains or what its purpose is. For example:

```
<meta name="Description" content="Missouri Web Standards and Guidelines will help Missouri web developers write accessible pages using best practices." />
```

3. The “refresh” tag will either update a page or redirect the user to another web page after a certain interval. This tag should not be used, since it can cause some accessibility issues and may disable the user's “back” button. Instead, use a server redirect to forward the user to the correct page without the obsolete page ever being displayed.

Example of a meta redirect:

```
<meta name="refresh" content="300">
<meta name="refresh" content="3, http://www.w3c.org">
```

The first example refreshes the current page after five minutes. The second example sends the user to a new page after 3 seconds.

Other meta tags available are:

```
<meta http-equiv="Expires" content="Mon, 20 May 2004 01:00:00
GMT">
<meta name="author" content="Missouri Department of Agriculture">
<meta name="copyright" content="&copy; 2004 DMD">
```

More information about meta tags is available at <http://www.w3.org/TR/WD-html40-970708/struct/global.html#h-7.1.3.2>.

- **5) Linking Cascading Style Sheets**

External CSS files are linked in the head of the page. There are two main methods of linking an external style sheet to a page: linking and “@import”.

Linking is recognized by all browsers:

```
<link rel="stylesheet" href="/path_to_css/default.css"
type="text/css">
<link rel="alternate stylesheet"
href="/path_to_css/hicontrast.css" type="text/css"
title="High Contrast">
```

The “rel”, “href” and “type” attributes are mandatory. “Rel” can be either “stylesheet” or “alternate stylesheet”.

“Alternate stylesheet” allows additional style sheets to be available if the browser supports stylesheet switching. At present, Mozilla-based browsers, such as Netscape, Firefox and Safari, and Opera support alternate style sheets. Internet Explorer does not.

The other way to link to an external style sheet is with an “@import” statement, which looks like this:

```
<style type="text/css">
<!--@import url("/path_to_css/default.css");-->
</style>
```

Most current browsers recognize the “@import” statement, but Netscape 4.x does not. This means you can use the “@import” to exclude certain styles from Netscape 4.x but allow other, more modern browsers to display the styles.

- **6) Naming conventions**

Guideline: Folder and page names should be “human-readable”, using actual words or abbreviations instead of numbers or random sequences.

As your agency web site grows and gets more complicated, using some standards naming conventions will help keep your site organized and easier to maintain.

Naming conventions work on three levels:

1. Folders/Directories
2. Pages/Files
3. Internal names

Folder names should, if possible, describe the contents of the files with a simple word or

abbreviation. This will help your maintainers and users navigate through the site. For example, the two main sections of the Missouri Department of Revenue site are for taxes and motor vehicles/drivers licenses. Their folder names are “tax” and “mvd1”, respectively.

Page names are more complicated, and have more guidelines:

Always make sure the file has an extension (“.htm” or “.jpg”, for example). The main page of every folder should be named “index.htm” or “default.htm”, depending on your server setup. Many search spiders simply look in a directory for available files. Many Web servers are configured to look for an "index.htm", "index.html", "index.shtml", or "index.cgi" file when simple directory calls are made. Users can reference just the domain and folder path to get the main page instead of trying to remember the specific page name. Use the “.htm” extension on all XHTML/HTML files. This extension is most common and is easier to remember than “.html”, “.shtm” or any other variation. If you are using a server technology such as JSP, PHP or ASP, use the appropriate file extension (.jsp, .php or .asp, respectively). The state server is case-sensitive. Use lower case names whenever possible. Avoid using spaces or special characters in your file name. Some search engines, browsers and even servers do not recognize the space or other special characters as legitimate file name characters. If you need a separator between words, use an underscore “_”.

Within your pages, if you use the following suggested id attributes for your <div> tags, you will find it easy to identify the different parts of your web page. If other agencies use the same naming convention, you will be able to substitute other state style sheets for your own.

<i>Id name</i>	<i>Description</i>
header	The entire header of the page
global-nav	Primary navigation, usually right below the header
search	Search form
sub-nav	Secondary navigation, usually in a sidebar
content	The main content of the page
features	Special features, such as a calendar or special event
news	News headlines
tert-nav	Third-level navigation, if you need it
footer	The footer of the page

Illustration 4 shows an example of how to use the id names.

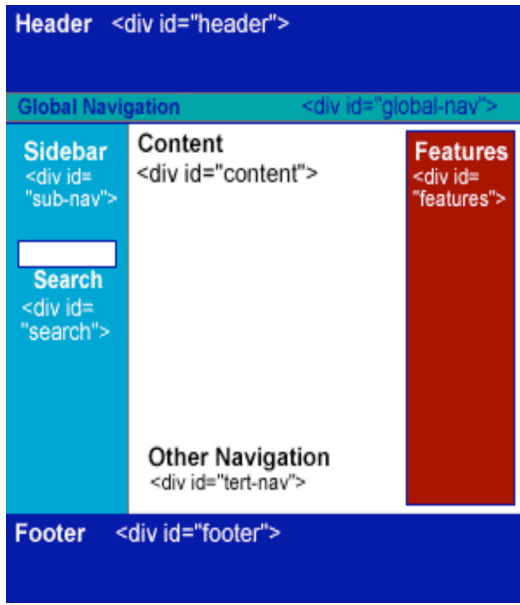


Illustration 4--div tags with ids

- 7) *Color and Graphics*

If you are starting to follow standards, you already have a well-formed text page with all your content, navigation and other features. But you still have a plain, black and white page.

Colors and graphics add life to your Web site. They add branding and consistency to your site. But they can trip you up and render your pages inaccessible and unusable.

- a. *Colors and Backgrounds*

1. Create a color palette for all the colors you will use in your site.

2. Use a style sheet to set your colors and background colors. Do not use tag mark-up.

- Bad: `<body bgcolor="blue">`
- Good: `...<style>body {background-color: #00f; color: #fff;}</style></head> <body>`



Illustration 5--Color Palette

3. Make sure your foreground and background colors offer enough contrast to be readable.

4. If you use an image in your background, make sure it is uncluttered and uses low contrast and saturation so the text stays readable.

5. While most computer graphic cards can render thousands or millions of colors, make sure your colors degrade gracefully if your user only has 256 colors.

6. If you declare a color in your style sheet, explicitly declare the background color as well. This will help you avoid embarrassing inheritance problems and will make your code easier to read.

- *b. Graphics and Images*

Graphics make your pages interesting, but in order to be accessible, you need to make sure that the graphics aren't necessary; someone who either cannot see the graphics, or someone that has turned images off on his browser, should still be able to understand what the graphic means.

The easiest way to do this is by using an “alt” attribute to attach a short text description to each image. For example, if your logo is also a link to your home page, the alt attribute would be:

```

```

Use the following tips for images and graphics:

1. **Use ALT attribute to attach text to a graphic** for non-graphical browsers or browsers with graphics turned off.
2. Keep image file size small. Use a program such as Fireworks or ImageReady to compress your images.
3. Images should take no longer than a few seconds to download using a 56K modem.
4. Save image files with the extension “.gif” and “.jpg” to be used in Web pages. Use GIF for black and white and for flat color graphics. Use JPG for continuous-tone images (photographs) at either medium or low and for gradations (medium).
5. Save images in interlaced format (GIF 87a or 89a format) for faster display. Fireworks or ImageReady will do this automatically.
6. Set mode as indexed color format to restrict the colors in the image palette to 256 or less.
7. Set resolution to 72 dpi. This is the default for GIF files.
8. Set palette to adaptive/diffusion to reduce dithering or color distortion by the browser.
9. Keep in mind many people may view your graphics on a screen that is smaller than 640 x 480 pixels in size. MSN TV is 560 pixels wide and does not allow side scrolling, and PDAs and other devices have smaller widths.

- *c. Animated Graphics*

Animated graphics draw your user's eye by motion. While this is a good thing for ads, it is not necessarily good for accessibility or usability. Blinking images can trigger epileptic seizures at worst and may distract from your page content at best.

The following are tips to consider when using animated GIFs or Flash.

- Animations that loop continuously can be annoying and can be distracting to the user.
- Make sure the animation stops after a certain number of loops or allow the user to turn the animation off.
- Keep the size of the image small.
- Add time between frames to control the speed of the animation.

F. PDF Files

Portable Document Format (PDF) files use a the Acrobat Reader to display documents in a browser. The PDF format basically captures the print instructions for a document and allows the document to be viewed just like it would be printed from its native application. Think of it as a “What You Print is What You Get.”

Your most common use for PDF files should be forms that your agency requires users to submit for licenses, applications, filings and other ways of submitting or requesting information. For instance, you can make tax forms available so the users can print their own copy instead of going to the post office or library to pick up a copy.

PDF files should be the exception rather than the rule for posting information on your web site. Your first choice for publishing information should always be HTML.

- ***1) Disadvantages of PDF files***

There are several reasons why you should limit your use of PDF files on your web site.

1. PDF documents are often inaccessible. While Adobe has made great strides in making sure that PDF files are readable with screen readers, PDF documents are still less accessible than plain text or HTML.
2. PDF documents require a plug-in to display the files. While most mainstream browsers either include the plug-in or make downloading it easy, Acrobat still has to open a separate process to display the page.
3. PDF documents are large. While you can keep the file size fairly small by creating your PDF files efficiently, they are still larger than text or HTML files and require longer to download.
4. Unless your source document is well-formed, the PDF document you create from it will have no navigation within the document.
5. You will generally lose all your site navigation when the user opens a PDF document. The only way to navigate back to the web site is through the “Back” button.

- ***2) Advantage of PDF files***

There is one main advantage to PDF:

Files that are *meant* to be printed work well in PDF, in particular forms, brochures, newsletters and other information that you intend for the user to print rather than view on line.

- ***3) Do's and Don'ts***

PDF should never be the first choice for web publication. It is easy for some content managers to think to themselves, “I need to get this Word document published quickly. I'll just convert it to PDF and post it.” This is usually always the wrong thing to do. While it make take a few minutes to convert the document to HTML and clean it up so it validates, the results will always load faster and be more accessible than using PDF.

You should not scan paper documents to convert to PDF if you are publishing to the web. All you will be doing is taking a picture of the document, and the file will be large,

inaccessible and will not scale properly if the user tries to enlarge the page.

If you do need to scan a document, use Acrobat's optical character recognition (OCR) function to convert the characters to text, then clean up any errors in the OCR.

Always work from the source document. If you are publish a form that State Printing creates for you, ask them for a PDF copy of the document so you can work from their original digital file.

If you allow users to submit a form through the web, use an HTML-based form that uses CGI to capture the data. It will be much smaller than creating fillable fields in a PDF document and running it through your CGI.

If you require a user to print the form and submit it, consider making all the form fields fillable so the user can fill out the form on-line before printing. This will make sure all the fields are readable.

- *4) Creating PDF documents*

You can create PDF documents from any application that uses a printer. If the program doesn't have a toolbar to create a PDF document, you can drag and drop the file onto the Acrobat Distiller to create the new document.

If you are creating a PDF document from a word processor, such as MS Word, make sure your original document is well-formed. Use headings, paragraphs and other structural tools to create different section of the document. When you convert the file to PDF, Acrobat will pick up the heading and create a navigate tree for easy navigation.

4. Usability

Usability is the measure of how users experience your site. It is different from usefulness, which would be a description of the content of your site and how useful it is for your target audiences.

Examples would be:

- < How a user knows what site he is on.
- < How well a user can learn and remember how to use the site.
- < How well a user can find specific information.
- < How well a user can negotiate interactive features.
- < How satisfied a user is with using the site.
- < How well errors are handled for the user.

Say, for example, you manage the State Parks site (<http://www.mostateparks.com/>) for the Department of Natural Resources. You can see from your logs that you get many hits for Elephant Rocks State Park.

How easy is it to find Elephant Rocks on the State Parks page? Is it easy to navigate to? Is it easier to type in “Elephant Rocks” in the search box and get it that way?

Now pretend you are considering exploring Missouri state parks with someone in a wheelchair. How easy is it to find out what state parks have wheelchair accessibility?

In this case, the information is fairly easy to find. But how many other sites have you been to where you just aren't sure where to find what you are looking for? How many sites have you visited where you weren't sure where to go next to complete a download or submit a form?

Usability requires you to know what your users want from your site, know what you want from your site, and know how to figure out how to make it happen. The rest of this section will discuss some of the steps you can take to make your site more usable.

A. Branding

Branding is a way of identifying your agency web site. Branding lets users know that they are on a Missouri government web site, and that the site is part of the Missouri state government web.

The usual method of branding is by logo. A logo is a graphic element that your agency uses to identify itself in print or on the web.



Illustration 6--Missouri State Home Page logo



Illustration 8--Department of Conservation web logo



Illustration 7--DNR logo

Standard: Each page must have some way of identifying which agency is responsible for its content, either with a logo or text identifier.

By using a combination of a graphic and text in a banner at the top of each web page, you let users know what agency site they are on and give them a way to recognize other pages that your agency publishes. You can turn the graphic into an icon or, if the logo is suitable, you can even make it into a button that you use in unordered lists. This can help reassure the user that they are getting information from the right agency.

B. Other ways of branding

Guideline: You should also brand your site by using consistent colors, fonts and layout.

1) Colors

A consistent color scheme helps remind users that they are still in your site. Consistent colors may be as simple as having a common background color or pattern for all your pages, even though you may use different accent colors to differentiate parts of your site.

Missouri Department of Revenue (<http://www.dor.mo.gov/>) uses a consistent template with the same background color, but they use a different sidebar and accent color

depending on the section that users are in. Tax pages are red, vehicle and drivers' license pages are green and administration pages are blue. The DOR identity is maintained through all these pages, but the user is made very aware if they inadvertently stray from a tax page onto a vehicle license page. This helps keep users on track for the information they seek.

2) Fonts

Using a consistent set of fonts is another way of branding your site. Some newspapers, like the Wall Street Journal or New York Times, use a distinctive font so that, even if you only see a clip of an item, you can tell which paper the article came from.

You can set your style sheet to accomplish the same sort of font brand identification. While you want to limit your set of fonts to those most commonly found on users' browsers, by using a particular font and font size for all your <h1>, a consistent font and size for your <h2>, <p> and other tags, users will gain a sense of familiarity with your site because the pages all have a similar look.

In this example, all the headers have the same font family, and all the paragraphs have a different family. The headers all have different sizes to emphasize their place in the document.

```
h1, h2, h3, h4 {font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;}
h1 {font-size: 1.4em;}
h2 {font-size: 1.2em;}
h3 {font-size: 1.1em;}
h4 {font-size: 1.0em;}
p {font-family: "Times New Roman", Times, serif;}
```

3) Layout

If you use a consistent layout, preferably by developing templates, your site will have a familiar feel to it. Users will know where to find different navigation tools, links, and page content because each of those elements are carried over from the page they just came from.

Using consistent colors, fonts and layouts gives your users a sense of familiarity with your site, and they will be encouraged to return in the future. Consistency also builds trust in your site, because users will know what to expect as far as your page appearance goes.

B. Navigation

Your site navigation scheme is the most important usability feature on your site. A usable navigation scheme will allow your users to find any information they need quickly and confidently.

There are many possible elements you can use to build your navigation scheme. The most common elements are global navigation, sidebars, breadcrumbs and site maps. There are many ways to use each of these elements, but the important thing is to remember to use them consistently across all your pages.

1) *Global navigation*

Global navigation contains the links to pages that helps users no matter what page on your site they are on. Good candidates for global navigation links include:

- < your home page
- < site map
- < search page
- < help page
- < jobs page
- < contact information
- < state home page.

All these items are either for general information or help the user use your site better.

The most common location for global navigation is tucked right under your header, where it will show up with your logo and other branding elements. Alternatively, these could be placed in a sidebar or in your footer. The important thing is to be consistent so users will know where to find these links.

There are several ways to use these items. For instance, you could have a simple link to your search page in global navigation, or you could have a text box to allow users to search directly from your global navigation. You could also have a fly-out menu that would show related links under each main link.

How many of these items you use is up to you as the web designer. As a rule, simpler is better, and many Missouri agencies use only a few, most often jobs, help and search links.

Examples:



Illustration 10--www.mo.gov header and global navigation



Illustration 9--DESE banner and global navigation

2) Sidebars

“Sidebar navigation” is really not the right term. A better description is “secondary navigation,” but sidebars are the most common form of secondary navigation.

Secondary navigation points the user to different areas of specific interest on your site. For instance, you may have pages for the general public, a section for regulated industries and another section that explains how your agency operates.

Guideline: Secondary navigation should be consistent throughout your site. While individual links may change, the look-and-feel should become familiar as the user navigates your site.

In the following examples, Agriculture uses fly-out menus to allow users to go deeper into the MDA site. DESE divides their menu into functional areas with popular links grouped under each topic.



Illustration 12--Agriculture Secondary Navigation



Illustration 11--DESE Secondary Navigation

Secondary navigation can also be included in a header or footer, like Natural Resources does in this example:

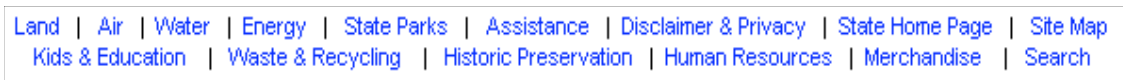


Illustration 13--DNR Secondary Navigation in a footer

3) *Breadcrumbs*

Breadcrumbs are a text-based reference to your site's structure. They show users where they are in the site and give a trail for them to follow to get back to a parent section.



Illustration 14—MDI Breadcrumbs

Breadcrumbs give users an option to using the Back button for navigation and can help some users negotiate your site more effectively.

While studies have not shown that users have any strong preference for using breadcrumbs for navigation, effective use of breadcrumbs can show the logical structure of your site and give clues to users for future searches.

C. Layout

As mentioned in the “Branding” section, having a consistent layout to your pages helps brand your site and makes it feel familiar to users.

A good layout will help your users use your site more effectively as well. When your header, navigation, content and footer are in the same place for each document in your site, users know where to look to find content, links, and other information.

While there are countless layout possibilities for your site, there are a few principles to keep in mind:

- < Pages should flow. Users instinctively look at the top left corner of the page when a page pops up. Your layout should help direct their attention to your content.
- < Put important content “above the fold.” Users should get the most important information on your page without having to scroll.
- < Use white-space in your layout. Putting some blank space around your content makes it more prominent and easier to focus on.
- < Make your page easy to scan. Users tend to scan pages instead of reading them. When you keep lines of text short (7-11 words), you make it easier for users to comprehend your content.
- < Put related items close together. For instance, DESE's navigation sidebar (Illustration 7) explicitly puts related topics in groups. Agriculture (Illustration 6) uses lines to separate their sidebar into department functions and department programs. Illustrations should appear near relevant text.

Too much text and too many images can actually make your page harder to use. Break different sections of your page up and make them as distinctive and uncluttered as you can. Users will then be able to focus on your core items, and use your navigation tools to find everything else.

There are a couple of simple tests you can do to evaluate a new layout. One is to resize your window to popular resolutions (1024x768, 800x600, 640x480) and see how the page fits. Does it justify to one side? Does it expand and contract to fit the window?

Another way is to look at your layout from across the room. If you look at your page from the hall outside your cube, what do you see? Is there good contrast? Do the colors work well together? Is the layout balanced?

Guideline: The simpler the layout, the better.

D. Page speed

Another element of usability is speed. How quickly does your page pop up in a browser. That depends on how much code your page contains.

Guideline: The smaller file size, the better.

What affects your page speed? Here's some common elements that add weight to you page:

- < the HTML code in the page itself
- < server-side includes
- < graphics
- < external scripts, like Javascript or CSS
- < multimedia, like sound, video or Flash

1) HTML

Make sure you don't have extra code in your page. If you have converted a document from Microsoft Office, for example, you can strip out all the formatting code that the conversion puts in and reduce your page size up to 80 percent.

Make sure you use CSS to format your HTML elements instead of using , <center> and other obsolete tags, and switch to a table-less format to reduce all the extra code needed for tables (remember, those <tr> and <td> tags with their attributes start bytes to your page pretty quickly).

It may help to think of it as a ratio of content to code: the higher the ratio (more content, less code), the better.

2) Server-side includes

Server-side includes are good things to use, but you need to keep in mind that each one adds weight to your page.

Guideline: Keep your server-side includes as lightweight as possible.

Keep comments out of your server-side includes. You don't need those comments repeated on every page. Instead, consider explaining what your SSIs do in your documentation for your site.

3) Graphics

Guideline: Make your graphics as small as possible.

No, don't resize all of them to 10x10 pixels. But use a graphics editor like Fireworks to optimize the file size.

For GIFs, look at how many colors your graphic actually uses. If you can reduce it to a 32 or 64-bit palette, your file size will decrease dramatically.

For JPGs, reduce the quality to 50 or 60 percent. This will reduce the file size without noticeably affecting the image quality on the web.

4) External scripts

If you are linking external scripts or CSS in your file, make sure each page you link from actually needs the external file. While these files can be cached, you can't be sure of any user's browser settings, so you don't know if unnecessary external files are slowing your page display.

Remember, too, that an estimated 10% of users may have Javascript turned off, or may not be able to use it at all. There are different versions of Javascript as well, and if you are using a Netscape or IE specific version instead of a cross-platform version, your users may not benefit from your scripting at all.

5) Multimedia

Sounds, video and Flash can make your page more interesting, but at a cost. Make sure you save your audio and video files in a compressed format and keep the files small.

Multimedia files require helper applications, which have to load in the background and may cause delays in your page.

And remember that many users don't have speakers available, or are non-visual. Any content presented in multimedia should have an alternate presentation for accessibility.

6) Check your page speed

Both the IE and Mozilla/Firefox Accessibility Toolbars include a link to <http://www.websitesiteoptimization.com>, a free service that measures your page speed.

The site looks at your web page, calculates the size of all the code, images, CSS, scripts and any other element and tells you how long a browser will take to download the page at various speeds.

You should check your pages against www.websitesiteoptimization.com during any redesign to see what page components may be slowing down your page.

E. Plug-ins

Almost any code that isn't HTML, XHTML, text, GIF or JPG requires some additional software for the browser to display it. The additional software is called a "plug-in." The most common examples are PDF files, which require Adobe Acrobat or Adobe Acrobat Reader to view, and Macromedia's Flash, which displays interactive content.

When you use these file types, you force the user's PC to open up the plug-in and download the file. As a rule, these files are larger than an equivalent HTML file, so the user has to wait for the helper program to open and then wait for the file to download. If the file is large and a user has only a dial-up connection, the user could have to wait several minutes to see the content.

Standard: If you use files that require a plug-in, you must provide a link to allow users to download and install the plug-in.

Standard: You must identify the type of plug-in needed to view the document before the user clicks on a link to view the document.

You cannot assume that users have Acrobat Reader, Flash, Microsoft Office or any other software on their computers. You can't even be sure what browser they may use, and whether that browser will help them with plug-ins. That's why you need to give them the link.

If you identify the type of plug in, or the file type of the link, you give users some choice in how they may want to view the document. If you have a video, for instance, users can decide whether they can use already-installed video software or if they want to download a new program.

Guideline: You should indicate the file size of any document that requires a plug-in.

If you clearly mark that a particular audio or video file is several megabytes, users can decide whether or not to click on the link and wait for the download. If you don't indicate file size, users may initiate a download and get frustrated if the file takes more than a few seconds to load.

Guideline: Avoid gratuitous use of plug-in technology.

Just because you can create a multimedia splash page doesn't mean that you should. Make sure you have a reason to use the technology you use. If the plug-in file doesn't add information or illustrate an example, think twice before you add that file to your site.

Splash pages done in Flash are almost never necessary on government sites. They may be appropriate on a marketing page, where users expect to be entertained and amazed, but government is generally here to inform, not entertain. If you want to amaze the users, make your site simple and useful.

Examples:

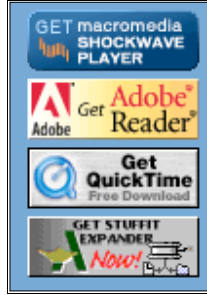


Illustration 16--
MDC plug-in links



Illustration 15--MDI Plug-in identification and file size

1) PDF, Flash

PDF files should be used when it is important to preserve the format of the document, like in a form that users must mail to your agency, or when it is assumed that the document will be printed, like a brochure. Even with documents designed for print, though, the content should be available on your site in HTML format.

Flash files are appropriate for specific applications, such as games or video, but should not be used as the sole means of presenting content. Flash is not always accessible, so you should make sure meaningful content is available in HTML format.

While some web development tools, like Dreamweaver, make it easy to create Flash buttons for your site, you should use HTML and CSS to create any presentation or rollover effects for your links.

2) Audio and Video

Audio and video files should be used with extreme caution. Both require hardware as well as software for users to hear or see the files, and you cannot assume that they have the capability to use the files.

For instance, many businesses and state agencies do not install sound cards or speakers on desktop PCs. Users cannot hear any audio files and only get to watch silent movies. Older systems may have card that cannot handle large files well, leading to skips in audio or jumpy video.

Audio/video files can be extremely large, and can take a long time to download on a dial-up system. This can be extremely frustrating for some users if they don't know how large a file is before they start to download.

Audio and video files can come in a variety of formats and no media player can display all formats. However, mp3 and mpeg (also known as mpg) files follow standards set by the Moving Picture Expert Group (MPEG—<http://www.mpeg.org/>) and are supported by most media players.

F. Resources

Usability First: <http://www.usabilityfirst.com/>

Usability.gov: <http://usability.gov/guidelines/index.html>

Sitepoint: <http://www.sitepoint.com/subcat/usability>

Web Pages That Suck: <http://www.webpagethatsuck.com/>

5. Accessibility

Accessibility means that users can use your site to get to your information. That's all. It means that you don't put up any barriers or impediments to getting to your information.

It doesn't mean that your web site is dull and colorless. It doesn't mean you can't have audio and video files. It doesn't mean you have to buy expensive software or hire experts to test each page of your site.

Accessibility just means that you don't code your pages in a way that keeps people from getting to your information.

The good news is that, if you have followed the standards and guidelines in this guide so far, you are eighty to ninety percent accessible already. We'll show you how to get the other ten or twenty percent in this section.

A. 508 Accessibility Standard

The 508 accessibility standards come from the federal Rehabilitation Act Amendments of 1998, which requires the U.S. federal government to make sure that any electronic or information technology they purchase will allow disabled citizens and employees to access or use the technology. Many states, including Missouri, have adopted these standards in state law.

While Section 508 covers everything from copiers to software, we are concerned with how it affects web pages.

B. WAI Standard

The other accessibility standard comes from the W3C, and is called the Web Accessibility Initiative, or WAI for short. It features a three-tiered approach to accessibility, with the first tier being most important, the second tier a little less important, and the third tier relegated to “nice to have” status.

You can think of the WAI standard as the official standard of the web, and as close to an international standard as we can get.

C. The Missouri Standard

Missouri has adopted the 508 standard in [Section 191.863](#) of the Revised Statutes, which states:

- “1. The [Assistive technology] council shall work in conjunction with the office of information technology to assure state compliance with the provisions of Section 508 of the Workforce Investment Act of 1998 regarding accessibility of information technology for individuals with disabilities.
2. When developing, procuring, maintaining or using information technology, each state department or agency shall ensure, unless an undue burden would be imposed on the department or agency, that the information technology allows employees, program participants and members of the general public access to and

use of information and data that is comparable to the access by individuals without disabilities. “

OIT has developed a standard to implement the statute. You can download the standard at: (http://oit.mo.gov/standards/ITGS0003_Missouri_IT_Accessibility_Standards.doc)

The Missouri standards closely follow the 508 standards on web development, with the exceptions of (b) and (p). The Missouri standards read as:

Web-based Intranet and Internet Information and Applications

- ⟨ A text equivalent for every non-text element shall be provided (e.g., via "alt", "longdesc", or in element content) except for captioning of audio information which shall comply with (b) of this section.
- ⟨ Captioning, video description or other equivalent alternatives for multimedia presentations, excluding live Webcasts, shall be provided in synchrony with the presentation, and in accordance with the following:
 - ⟨ Captioning shall be provided for multimedia presentations that contain speech or other audio information necessary for the comprehension of the content in accordance with the schedule established in Paragraph (c) under Video and Multimedia Products.
 - ⟨ Video description shall be provided for multimedia presentations that contain visual information necessary for the comprehension of the content, in accordance with the schedule established in Paragraph (d) under Video and Multimedia Products.
 - ⟨ Live Webcasts that contain speech or other audio information necessary for the comprehension of the content, shall be captioned in accordance with the following schedule with priority given to content of statewide importance and events that do not provide the opportunity to request individual accommodations.
 7. By June 30, 2005, 10 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.
 8. By June 30, 2007, 25 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.
 9. By June 30, 2009, 50 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.
- ⟨ Web pages shall be designed so that all information conveyed with color is also available without color, for example from context or markup.
- ⟨ Documents shall be organized so they are readable without requiring an associated style sheet.
- ⟨ Redundant text links shall be provided for each active region of a server-

side image map.

- < 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.
- < Row and column headers shall be identified for data tables.
- < 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.
- < Frames shall be titled with text that facilitates frame identification and navigation.
- < Pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz.
- < A text-only page, with equivalent information or functionality, shall be provided to make a Web site comply with the provisions of these standards, when compliance cannot be accomplished in any other way. The content of the text-only page shall be updated whenever the primary page changes.
- < 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.
- < 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 the standards set forth under “Software Applications and Operating Systems”, paragraphs (a) through (l) contained within this document.
- < 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.
- < A method shall be provided that permits users to skip repetitive navigation links.
- < When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required.
- < Contact information for issues related to accessibility shall be provided on each entry page.

Item (b) goes into much greater detail about making multimedia accessible and provides a time line for compliance than 508 does..

Item (p) is a new item not found in 508, and calls for an accessibility contact link on each home page.

D. Missouri Accessibility Implementation

Ok, so how do you implement these standards?

If you've followed the guidelines in the previous sections, you have done the hardest part towards accessibility, and you'll find that you've implemented most of these already.

We will assume that you are using either Internet Explorer 6 or Mozilla Firefox to check your pages. You will need to download either the AIS Web Accessibility Toolbar for IE (<http://www.nils.org.au/ais/>), or the Web Developers Toolbar for Firefox/Mozilla (<http://www.chrispederick.com/work/firefox/webdeveloper/>) and Checky (<http://checky.sourceforge.net/#Install>). If you haven't already installed these on your browsers, do it now. We'll wait. You'll thank us later.

Ready? Let's go through the standards, one by one, and figure out how to hit them.

(a) A text equivalent for every non-text element shall be provided (e.g., via "alt", "longdesc", or in element content) except for captioning of audio information which shall comply with (b) of this section.

- *Alt-text Guidelines:*

Keep wording simple

Bad: alt="DMD's logo and link to DMD home page"

Good: alt="DMD home"

Describe function of graphic rather than what it is or looks like.

Bad: alt="right-pointed triangle graphic"

Good: alt="next"

Include punctuation or spaces at the end of the alt-text of images used as links. Remember that screen readers look for punctuation to signal pauses and ends of sentences.

Of course, if you have been redesigning your site with standards, you won't need any spacer gifs because your site is laid out with CSS, but if you need to use a "spacer.gif" for layout, the alternate text should be alt="" so screen readers don't waste time trying to interpret the text.

If you use graphics for buttons or links, the alt text should indicate where you are linking to and not the word "Link" or "button." There is no need to use Link attributes since the href and alt attributes are both used.

Bad: ``

Good: ` `

- *Longdesc*

A longdesc attribute is a link to a file that provides more information about the image.

- *Title*

A title attribute can be applied to nearly any tag, including the tag. It just gives the user a caption to go with the element. In visual browsers, the title attribute usually shows up as a "tool tip" pop-up message. In audio browsers, the title may just be read aloud.

This attribute can be used to provide additional information to an image.

- “[D]” link

Since not all browsers can render the longdesc attribute, it is helpful to add a link immediately after the image using “[D]” to link to the description page.

Code example:

```
<a href="2004employment.htm">[D]</a>
```

where 2004employment.htm reads:

```
<table class="cspace">
<caption>Missouri Annual Average Employment Statistics</caption>
<thead>
<tr>
<th scope="col">Year</th>
<th scope="col">Labor Force</th>
<th scope="col">Employment</th>
</tr>
</thead>
<tbody>
<tr>
<th scope="row">2003</th>
<td>3,020,592</td>
<td>2,850,466</td>
</tr>
</tbody>
</table>
```

- *Testing for alt text*

4. Use your Web Developer or AIS toolbar to turn off images. Do you see any text where the images were?
5. Mouse over the images. You should see a “tooltip” in Firefox if there is a “title” attribute present.

(b) Captioning, video description or other equivalent alternatives for multimedia presentations, excluding live Webcasts, shall be provided in synchrony with the presentation, and in accordance with the following:

4. Captioning shall be provided for multimedia presentations that contain speech or other audio information necessary for the comprehension of the content in accordance with the schedule established in Paragraph (c) under Video and Multimedia Products.
5. Video description shall be provided for multimedia presentations that contain visual information necessary for the comprehension of

the content, in accordance with the schedule established in Paragraph (d) under Video and Multimedia Products.

6. Live Webcasts that contain speech or other audio information necessary for the comprehension of the content, shall be captioned in accordance with the following schedule with priority given to content of statewide importance and events that do not provide the opportunity to request individual accommodations.

By June 30, 2005, 10 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.

By June 30, 2007, 25 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.

By June 30, 2009, 50 percent of all live Webcast hours transmitted during the preceding twelve months by a State department or agency must incorporate captioning.

This is one case where the Missouri standard goes beyond the 508 standard by giving a time line for accessible live webcasts. So how do we do this?

7. For audio webcasts, make sure there is a transcription available, either from a script or by close-captioning the webcast as it airs.
8. For video web casts, provide accessible close captioning of the audio as well as a description of any activity that a visually-impaired user may need to know.
 - Place an anchor to a page with a text transcript or description of the clip right next to the anchor for the clip.
 - If the user has requested a text-only page, replace all URL references to the clip with URL references to the text transcript or description.

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

There's a couple of aspects to this standard.

First, avoid using words like “click the green button to continue. Click the red button to reset the form.” A color-blind person may only see a couple of gray buttons. Instead, label each button “continue” and “reset” so there is a textual clue to follow.

Second, make sure background patterns and colors contrast well with the text color and select colors that will make pages easy to read by people with color blindness.

- *Testing for color issues*

10. Go to <http://colorfilter.wickline.org/> or <http://www.vischeck.com/vischeck/vischeckURL.php> and enter your URL.

11. In IE, use the “colour” tab in your AIS toolbar to check your page in grayscale.
12. In Firefox, right-click on the page. Select “Checky>>WAI/508>>Color Blind Web Page Filter. Select a type of color blindness to see your page rendered as a color-blind person may see it.

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

If you've started your web page as a plain text page, then applied styles to add colors, images, fonts and positioning, you won't have much of a problem here.

- ***Testing***

Turn off style sheets with the Web Developer or AIS toolbar or view the page in a text browser to see how it displays.

(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.

Ah, image maps. They let you point to various parts of an image and click a link. They are a great way to carve up a map of Missouri into regions, counties, cities or anything else you can think of.

Let's set a few rules for them.

- Server-side image maps should be avoided. Back in the early days of the Internet, browsers did not have the capability of interpreting an image map, so all the link coordinates had to be generated server-side. Naturally, there was little thought of accessibility. When a user mouses over the image map, they get a set of coordinates, but not an actual link that they can see in the status bar. Plus, users have to use a mouse; there is no natural text equivalent for the link.
If you have to use a server-side image map, you will need to include a separate text list of links on the page to give the users another way to access the information.
- Client-side image maps are part of a web page and are interpreted by nearly all browsers. They are easy to create and make accessible. There is virtually no kind of shape you cannot make with a client-side map. If you have to use an image map, use this kind.
- Only use an image map when there is no suitable text-based solution. For instance, to let users pick a region of Missouri on a map. Don't use it to make your navigation bar at the top of your page; that is what text and CSS are for.

- ***Image map hot spots***

If you have image maps, your main image should have an alt attribute. You also need an alt attribute for each hot spot link.

Example:

```

```

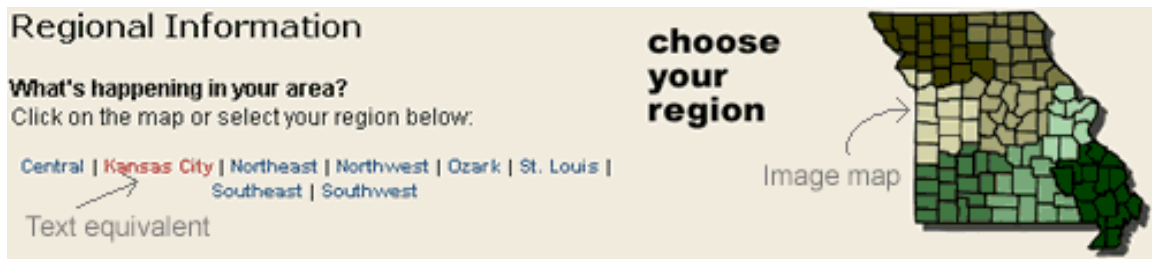
```

usemap="#countymap" alt="Map of Missouri counties"/>
<map id="countymap">...
<area shape="poly"
coords="188,221,188,217,143,216,143,227,143,251,144,252,187,252,188,221"
href="/cgi-bin/hoAvail.cgi?sr=1&cnty=83" title="Henry County" alt="Henry
County" />...
</map>

```

It is also a good idea to have a text list of links for your image map. For instance, if you have five regions in the state, show the text equivalent somewhere on the page.

Illustration 17--Image map with text equivalent



- *Testing*

Turn off images with the Web Developer or AIS toolbar. Try to navigate.

(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.

Data tables are one of those things that you really have to work at to make accessible. Fortunately, once you get the hang of it, it doesn't take too long to mark the table up to make it accessible to any modern browser.

There's several things to keep in mind with any table:

6. We are talking tables for data here, not layout tables. You may end up with a layout table in your site, but we hope you'll try to avoid nested-table layouts. Try to limit your use of tables for layout by using text and CSS. This is not always possible, but

dare to dream.

7. A data table should be marked up semantically to identify its various parts. This is half the battle of accessibility.
8. This standard is the bare minimum for table accessibility. We'll show you how to make it even more accessible.
9. If you've tried everything else (or are just lazy or didn't bother to read the rest of this guide) and have had to resort to a layout table, very little of this section is going to apply to that layout table. You should keep your tables very simple, since you *don't want* the layout tables to convey any information.

Let's take a look at a basic data table in Illustration 2:

Player	G	AB	R	H	HR	RBI	BB	K	AVG
Albert Pujols	154	592	133	196	46	123	84	52	.331
Scott Rolen	142	500	109	157	34	124	72	92	.314
Tony Womack	145	553	91	170	5	38	36	60	.307
Jim Edmonds	153	498	102	150	42	111	101	150	.301
Total	148.5	2143	435	673	127	396	293	354	.314

- *Structural part of data tables*

There are usually four structural parts of a table: caption, head, foot (if the table has some summary figures), and body. These are coded as `<caption>`, `<thead>`, `<tfoot>` and `<tbody>`. These components are part of the HTML/XHTML specification, so it is certainly alright to use them, even though your page may validate and be accessible without some of them.

- **<caption>**

The `<caption>` is just a title for your table, and is important for accessibility. When you use the `<caption>` tag, you explicitly associate the table's title with the table. Example:

Bad: `<p>2004 Cardinal Batting Averages</p><table><tr>...`

Good: `<table><caption>2004 Cardinal Batting Averages </caption><tr>...`

While the `<caption>` resides within the `<table>` tag, it usually does not share the table's default styles, like background or border. Of all the structural table tags, this one is most useful for accessibility since it tells the user what is in the table.

- **<thead>**

`<thead>` identifies the column headers in your table. While not always necessary in a simple table, `<thead>` adds structural markup to your table and lets the browser know that the tags in the `<thead>` are important markers. If you have a complex table with multiple sections, you may have several different `<thead>` sections.

In our example above, the `<thead>` section looks like this:

```
<thead>
  <tr>
    <th scope="col">Player</th>
    <th scope="col"><abbr title="Games Played">G</abbr></th>
    <th scope="col"><abbr title="At Bats">AB</abbr></th>
    <th scope="col"><abbr title="Runs Scored">R</abbr></th>
    <th scope="col"><abbr title="Hits">H</abbr></th>
    <th scope="col"><abbr title="Home Runs">HR</abbr></th>
    <th scope="col"><abbr title="Runs Batted In">RBI</abbr></th>
    <th scope="col"><abbr title="Walks">BB</abbr></th>
    <th scope="col"><abbr title="Strikeouts">K</abbr></th>
    <th scope="col"><abbr title="Batting Average">AVG</abbr></th>
  </tr>
</thead>
```

- **<tfoot>**

`<tfoot>` comes, somewhat counter intuitively, before the body of the table. `<tfoot>` is a way of marking the summary data of a table, like the “totals” row (or rows). This allows browsers to display the data in `<tfoot>` before the rest of the page fully loads. `<tfoot>` is displayed at the bottom of the table.

Many simple tables may not have a “totals” row, and may not need to use a `<tfoot>`. Complex tables, however, may have one or more `<tfoot>` sections.

In our example, the `<tfoot>` section looks like this:

```
<tfoot>
  <tr>
    <th>Total</th>
    <td>148.5</td>
    <td>2143</td>
    <td>435</td>
    <td>673</td>
    <td>127</td>
    <td>396</td>
    <td>293</td>
    <td>354</td>
    <td>.314</td>
  </tr>
</tfoot>
```

- **<tbody>**

<tbody> is structural markup to identify the main data items of the table and follows <tfoot>. With complex tables, it may be possible to have several <tbody> sections, although if you find yourself in that situation, you may want to consider breaking the data out into separate tables.

If you are doing any coding to generate your tables, using these tags will help your programmers identify each section of the table and make it easier to code.

Here's the <tbody>:

```
<tbody>
  <tr>
    <th scope="row">Albert Pujols</th>
    <td>154</td>
    <td>592</td>
    <td>133</td>
    <td>196</td>
    <td>46</td>
    <td>123</td>
    <td>84</td>
    <td>52</td>
    <td>.331</td>
  </tr>
```

...

```

<tr>
  <th scope="row">Jim Edmonds</th>
  <td>153</td>
  <td>498</td>
  <td>102</td>
  <td>150</td>
  <td>42</td>
  <td>111</td>
  <td>101</td>
  <td>150</td>
  <td>.301</td>
</tr>
</tbody>

```

- *Making the table accessible*

Now that we've identified our table parts and put an accessible caption on the table, let's look at making all those columns and rows mean something.

- **Table headers**

You've probably done something like this in the past in order to label the columns in your tables:

```

<tr>
  <td><center><b>Player</b></center></td>
  <td><center><b>Games</b></center></td>
  <td><center><b>At-bats</b></center></td>
  <td><center><b>Hits</b></center></td>
  <td><center><b>Home Runs</b></center></td>
  <td><center><b>RBI</b></center></td>
</tr>

```

What's wrong with that? Well, there's nothing in the code to tell the browser or the user what the tags really mean. As far as the code goes, this is all data. The only way we would know that these are column headers is by context, and that is getting away from usability and accessibility.

So how should we do it? By using a table header `<th>`, like this:

```

<thead>
  <tr>
    <th>Player</th>
    <th>G</th>

```

```

    <th>AB</th>
    <th>R</th>
    <th>H</th>
    <th>HR</th>
    <th>RBI</th>
    <th>BB</th>
    <th>K</th>
    <th>AVG</th>
  </tr>
</thead>

```

When we use <th>, we identify those cells as headers for the data that follows. Most browsers will automatically center and bold the <th> tag as well, so you don't have to add additional styling or tags.

Now that we've added headers, we need to tie those headers in to the data columns. We will do this with a "scope" attribute. This tells the browser that the cells within the scope of that header are related.

```
<th scope="col">Player</th>
```

will associate every player name in that column with the caption "player".

You can also use <th scope="row"> on each line of data to identify a header for a row of data. That way, you create a cross-reference for each cell with a column and row.

Guideline: Use table headers <th>, captions <caption> and scope <th scope="col"> to define data tables. Example:

```

<table>
  <caption>Phone List</caption><tr>
    <th scope="col">Last Name</th><th scope="col">First Name</th><th
    scope="col">Number</th></tr>
  <tr><td>Boeckman</td><td>Debbie</td><td>(573) 555-8400</td></tr>
  <tr><td>Strange</td><td>Lanie</td><td>(573) 555-2500</td></tr>
</table>

```

- *More on tables*

If you have more complex tables, such as tables with multiple sections or headers that span multiple rows, accessibility is a little more complicated. You can find more information on tables at these websites:

- <http://www.w3.org/TR/REC-html40/struct/tables.html>
- <http://www.w3.org/TR/2000/NOTE-WCAG10-HTML-TECHS-20000920/>
- <http://www.webaim.org/techniques/tables/>
- <http://www.mcu.org.uk/articles/tables.html>

- <http://www.yourhtmlsource.com/tables/tablesaccessibility.html>
- http://www.ferg.org/section508/accessible_tables.html

Guideline: Do not incorporate these tags in layout table.

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

Guideline: Avoid using frames. Use a server-side includes to create headers, menus and other items that are common to each of your pages.

Frames are a method to include content from several sources in one screen on a browser. The most common elements are sidebars, headers and footers. These are all displayed in separate windows in frame in the browser window.

For the most part, any element you would normally put in a frame would be better used in a server-side include. This keeps all the elements in a single page instead of in separate pages in a single frame.

Frames can also cause confusion with navigation. When a user clicks on a link in a frame, the URL in the address bar does not change. This makes it very difficult to link to items in a frame.

Standard: If you have to use frames, each frame must be given a title attribute.

If the purpose of the frame is not apparent from the title, an alt or longdesc attribute should be included.

Standard: Use <noframe> to identify content for screen readers to read.

For specific techniques for making frames accessible, see the W3C guidelines at <http://www.w3.org/TR/WCAG10-HTML-TECHS/#frames>.

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

Flickering pages or images are distracting and can trigger certain types of epileptic seizures. Avoid using Javascript or DHTML effects that make the page flicker. When possible, avoid animation.

Do not use the <blink> and <marquee> tags. In addition to being browser-specific, non-standard code, <marquee> is often read by a screen reader one letter at a time as it appears on the screen and <blink>, well <blink> blinks, which is the same as a flicker.

(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.

If you have followed the standards and guidelines, you won't need to make a separate accessible text-only page. That's the whole idea behind standards.

If your page consists of image map navigation, for instance, you may have to create a separate page. If more than one of your pages are image maps, though, you will find that it takes a lot of work to maintain two versions of each page.

Guideline: If you have to create a separate text-only page, start thinking about how you can revise that page to make it 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.

While server-side scripting poses no problem, client-side scripting, Javascript and event handlers may not be usable to screen readers, text browsers or browsers with scripting disabled.

This is particularly true of some drop-down menus, where the user does not have a “Go” button to click.

- *Testing*

Disable scripting with the Web Developer or AIS toolbar and refresh to see if you can still read and navigate the page. If you can't, come up with a <noscript> alternative to the script.

(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 the standards set forth under “Software Applications and Operating Systems”, paragraphs (a) through (l) contained within this document.

If you want to use PDFs, Flash, Shockwave, Word or any other plug-in, make sure you give the user a link to get the plug-in. Make sure the plug-in is accessible. If the application is not accessible, be sure to provide an alt-text or longdesc tag to describe the content. If you are using Flash, use <noembed>.

(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.

Use the <label> tag to associate form labels with their form fields. Example:

```
<p><label for="email">E-mail address: </label>
<input type="text" id="email"></p>
```

You can also use the <label> tag to wrap descriptive text with a form field. Example:

```
<p><label>First Name
<input type="text" name="firstname"></label>
<label> Last Name
<input type="text" name="lastname"></label></p>
```

Labels are especially important for check boxes and radio buttons, since they let the user select the label text to toggle the button. This is a great help for users with mobility difficulties.

Add tab order such as `<input tabindex="1">` to forms to make the fields flow more smoothly with keyboard use instead of a mouse.

Use `<fieldset>` and `<legend>` to separate different parts of your form.

Sample code for an accessible form:

```
<fieldset>
  <legend>Policy Information</legend>
  <label for="covtype">Coverage Type: </label>
  <input type="text" name="covtype" id="covtype" tabindex="1">
</select>
  <label for="produc">Producer of Account: </label>
  <input type="text" name="produc" id="produc" value="" tabindex="2" />
  <label for="policy">Policy Number: </label>
  <input type="text" name="policy" id="policy" value="" tabindex="3" />
</fieldset>
<fieldset>
  <legend>Effective Date:</legend>
  <label for="effmon">Month:</label>
  <select name="effmon" id="effmon" tabindex="4">...
```

More information on forms is available at <http://www.w3.org/TR/html401/interact/forms.html>

If you have difficulties making your form accessible, provide a form that can be downloaded then mailed or e-mailed, or a phone number someone can call for assistance.

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

Since screen readers read every link on the page in order, most users will have to listen to all of your links before they hear any of your content. If your pages are like most, this means the user will hear all the links in your header and sidebar for each page.

Instead, use hidden links to allow the user to skip the menus and go straight to the content. Here's how:

```
...
</head>
<body>
<p class="none">[<a href="#content" title="Skip to content">Skip to content</a>]</p>
...[header][sidebar]...

<div id="content">
<p>This is the content part of the page.</p>
```



```
</div>
```

Your CSS would look like this:

```
.none {  
width:0;  
height: 0;  
overflow: hidden;  
position:absolute;  
}
```

Some people have suggested using “display: none;” to hide the code, but this ends up hiding the code for any browser that reads CSS, like some screen readers. The method demonstrated above doesn't keep the text from being read, it just hides the visual display, so the link stays accessible. (A tip of the hat to DESE, who discovered this bit of CSS. Thanks, Lainie!)

- *Testing*

Turn off the style sheet with the Web Developer or AIS toolbar and see if you can skip your navigation elements to get to the content.

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

Timed response may be used on login user sessions, forms, refresh and redirects. Users with disabilities generally take longer to navigate through a page, and may need more time.

Imagine taking a survey or quiz when suddenly the page auto-refreshes and wipes out all your answers. Or a slide show that goes on to the next slide before you finished reading the first one.

If you have a page that does an auto-refresh or a redirect, give enough time for a user to read your message and act (or not act). Consider a 10-second or longer pause, and make sure you put a link on the page to the redirect location so the user can click there if she just can't wait that long.

Example:

```
<meta http-equiv="refresh" content="10" url="newpage.htm" />  
...  
<p>This page has moved to <a  
href="newpage.htm">http://oa.mo.gov.dmd/newpage.htm</a>.</p>  
<p> You will be automatically redirected to that page in 10 seconds.</p>
```

(q) Contact information for issues related to accessibility shall be provided on each entry page.

This item is unique to Missouri. Your home page and main pages should have a link to

contact someone at your agency regarding any accessibility issues. This link could go to the web master, accessibility officer, or any other designated person in your agency.

E. Other accessibility points

If you have followed the guidelines and standards in this manual, you'll recognize these guidelines. If these items are new to you, you didn't read closely enough back in the beginning. Start again.

1) Links

Avoid using the phrase “click here” in your links. Use descriptive text within your normal sentence structure for your links.

- Bad: “To review the web guidelines on-line, [click here](#).”
- Good: “Review the [web guidelines on-line](#).”

If you cannot find a suitable phrase in your text, you can add a title attribute to your link. This will show up as a “tool tip” box in a browser, and should be read aloud by a screen reader.

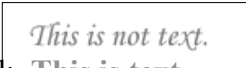
If you have a series of links that run together, like in a toolbar, make sure you use a space or a symbol to separate the links. Some older screen readers stumble over consecutive links without a separator.

You can hide the separator from a visual browser with CSS, using the same code example used in item (o) of the Missouri Accessibility Standards.

2) Image text

Guideline: You should avoid using graphics to display text.

This means that, instead of using Javascript rollover buttons for your navigation links in a sidebar, you should use text and CSS to display the link in its various states.

- Bad: 
- Good: **This is text.**

If you need to add text as part of a graphic, be sure that the text is available in an alt or longdesc attribute.

CSS rollovers are simple, as long as you remember to put the link styles in the right order:

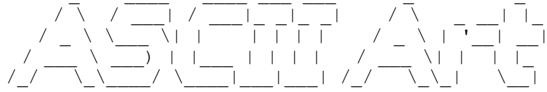
- a:link
- a:visited
- a:hover
- a:active

3) ASCII art and emoticons

Punctuation and symbols add meaning to your documents, but when you use them outside of a sentence, they lose meaning and can cause confusion.

Graphics created with punctuation marks and text symbols have no context and can confuse screen readers. While ASCII art is usually found in e-mail signatures, it can also be found on the web.

Example:



This example shows the word “ASCII Art” made up entirely of dashes, underscores, slashes and backslashes. While a sighted person may see a word there, a screen reader may see this as “hyphen space space hyphen hyphen hyphen space space, etc.”

Emoticons are characters that try to add meaning and nuance to a text message. The most popular example is the “smiley”, a colon, hyphen and closed parenthesis “:-)”. While these can add some context to an e-mail or text message (like, “Don't take this too seriously.”), an emoticon doesn't have any semantic meaning to a screen reader.

This really should not be much of a burden for a state web developer. Think about it. Do you really want to have content on your site that is so ambiguous that you have to tell the user that you are just kidding?

F. Resources

Section 508: <http://www.section508.gov/>
<http://www.access-board.gov/sec508/508standards.htm>

Accessibility book: “Building Accessible Websites” by Joe Clark
<http://www.joeclark.org/book/>

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

Dreamweaver Accessibility Extension:
http://www.usablenet.com/frontend/508as_entry.jsp