

Accessible Web Design

It is essential that the design of web sites makes them accessible to all users, regardless of disability or choice of browser. This Guide explains the benefits of accessible design, and techniques for writing accessible websites. You will need a web browser open to view some of the demonstrations.

Contents

- 1 About accessibility
- 2 Finding Accessibility Problems
- 3 Fixing accessibility problems
- 4 Other accessibility issues
- 5 Useful tools
- 6 Summary

Document code: **Guide 168**
Title: **Accessible Web Design**
Version: **1.1**
Date: **October 2003**
Produced by: **University of Durham Information Technology Service**

Copyright © 2006 University of Durham Information Technology Service

Conventions:

In this document, the following conventions are used:

- A typewriter font is used for what you see on the screen.
- A **bold typewriter font** is used to represent the actual characters you type at the keyboard.
- A *slanted typewriter font* is used for items such as filenames which you should replace with particular instances.
- A **bold font** is used to indicate named keys on the keyboard, for example, **Esc** and **Enter**, represent the keys marked Esc and Enter, respectively.
- A **bold font** is also used where a technical term or command name is used in the text.
- Where two keys are separated by a forward slash (as in **Ctrl/B**, for example), press and hold down the first key (**Ctrl**), tap the second (**B**), and then release the first key.

Contents

1	About accessibility	1
1.1	What is accessibility?	1
1.2	What is usability?	1
1.3	Legal Requirements	1
1.3.1	Quotes from SENDA	1
1.3.2	Maguire v Socog	2
1.4	Browser compatibility	2
1.5	Search Engines	2
1.6	University Policy	3
1.7	The Web Accessibility Initiative	3
1.7.1	Priority 1 recommendations	3
1.7.2	Priority 2 recommendations	3
1.7.3	Priority 3 recommendations	3
1.8	Other terms used in this Guide	4
2	Finding Accessibility Problems	4
2.1	Examples of inaccessible pages	4
2.2	Tools for finding accessibility problems	5
2.2.1	Web Browsers	5
2.2.2	Checking Tools	5
2.3	Identifying accessibility problems	5
2.3.1	Some of the accessibility problems with the example site	5
3	Fixing accessibility problems	6
3.1	Accessible text	6
3.1.1	Example of rewriting text	7
3.2	Accessible Graphics	8
3.2.1	Using Dreamweaver to make accessible images	8
3.2.2	Text Equivalents	8
3.2.3	Examples of 'text equivalents'	10
3.3	Accessible Tables	10
3.3.1	Using Dreamweaver to add header information	11
3.3.2	Example of adding headers to tables.	11
3.4	Accessible use of colour	11
3.4.1	Partial specification of colour	12
3.4.2	Specification of colour in Dreamweaver	12
3.4.3	Examples	12
3.5	Scripts, Java applets, Flash and similar technologies.	13
3.6	Frames	14
4	Other accessibility issues	14
4.1	Structural Markup	14
4.2	Validation	15
4.3	Navigation	15
4.4	Other areas to consider	16
4.4.1	Tables for layout	16
4.4.2	Forms for user input	16
4.4.3	Colour contrast	16
5	Useful tools	17
5.1	Browsers	17
5.2	Validators	17

5.3	Accessibility Checking Tools	17
5.4	Other tools	18
5.4.1	Specialist accessibility checking assistants.....	18
5.4.2	Link checkers.....	18
5.4.3	Tidy.....	18
6	Summary	18

1 About accessibility

1.1 What is accessibility?

A page is accessible if all users are able to access the document and read the content.

For a site to have good accessibility, users should be able to access it without needing to change browser (or install a new one, or move to a different computer with it installed), or adjust browser settings.

A well-designed site will work in any browser with any reasonable settings (clearly, sites are not expected to work in browsers where the user has set the colours to black text on a black background and prevented the author from overriding this).

1.2 What is usability?

Many accessibility improvements also make it easier for everyone to use the site. For example, if a site has an illogical navigation structure, then everyone will find it difficult to use the site (though those with a good memory and visualisation abilities will find it slightly easier). Improving the navigation structure helps everyone, but also especially helps people with poor memory.

There are also pure usability problems such as broken links, which affect everyone equally.

1.3 Legal Requirements

Under the Disability Discrimination Act 1995 it is illegal to unfairly discriminate. Legal opinion is that providing an inaccessible website would be unfair discrimination and therefore illegal.

The Disability Discrimination Act protects employees from discrimination and prohibits discrimination in the provision of goods and services. The Special Educational Needs and Disabilities Act 2001 (SENDA) extends the Disability Discrimination Act to students and applicants. Both Acts can be found through <http://www.hmsso.gov.uk/>

The Journal of Information, Law and Technology has an article discussing the applicability of the Disability Discrimination Act to websites: <http://elj.warwick.ac.uk/jilt/01-2/sloan.html>

1.3.1 Quotes from SENDA

"...not to treat disabled students less favourably, without justification, for a reason which relates to their disability..."

"...to make reasonable adjustments to ensure that people who are disabled are not put at a substantial disadvantage compared to people who are not disabled in accessing further, higher and Local Education Authority-secured education..."

1.3.2 Maguire v Socog

This case under the Australian equivalent to the Disability Discrimination Act was brought against the Sydney Organising Committee for the Olympic Games alleging that their website was inaccessible. It was decided that compliance with the Level A recommendations of the Web Accessibility Initiative was sufficient for compliance with that legislation. There is currently no case law in the UK, but legal opinion suggests a similar outcome is probable.

http://www.humanrights.gov.au/disability_rights/decisions/comdec/1999/DD000150.htm

1.4 Browser compatibility

There are hundreds of different browsers on the web, in a variety of different display media, from the common Netscape and Internet Explorer on Windows and computer monitors, to text-mode browsers, speech browsers and mobile phones.

Statistics such as "98% of users use Internet Explorer" are untrue. In the University, there is a roughly even split between versions of Netscape and versions of Internet Explorer, with a wide range of other browsers making up over 15% of page hits.

In the wider world, examination of web server logs suggests that between 70% and 85% of page hits are from browsers claiming to be Internet Explorer. There are several reasons that this cannot be used as an estimate of the proportion of users, however:

- Proxy and cache servers mean that users can view pages without making a request to the server the page is held on.
- Many browsers claim to be Internet Explorer (or less commonly Netscape Communicator) to access sites that reject users using other browsers.
- Speech browsers such as IBM Home Page Reader use Internet Explorer to make the connection to the website and download the page, before processing it themselves. Therefore, they appear in the web server logs as Internet Explorer, but display the page entirely differently.
- It is an unreasonable assumption that the number of users of Internet Explorer is 83% of the user population simply because 83% of hits are from Internet Explorer - users of other browsers may statistically spend more or less time on a site. On a site inaccessible to browsers other than Internet Explorer, users of other browsers will generate very few hits in proportion to their numbers.

A good summary of the inaccuracies of web statistics is at:

<http://www.analog.cx/docs/webworks.html>

1.5 Search Engines

Search engines have very limited capabilities for browsing. They cannot understand images unless an alternative text representation of the image is provided, most do not understand frames or follow links other than standard `<a>` tags (normal hypertext links instead of Javascript or Flash).

It is also impossible for a search engine to understand the content of a page and know which words are the keywords, unless the HTML code is written with structure instead of formatting in mind.

The quality of sample text in a search engine result will also depend to an extent on the accessibility of the content.

Accessible websites are easier for search engines to deal with, and their entry on search results pages usually has better content. On inaccessible sites that provide a 'text-only' version, the 'text-only' version often has a *higher* search engine rank than the normal version.

1.6 University Policy

The University has a policy on web page accessibility. Any site providing official university, department or college information to students, staff, or the public is covered. The summary of the policy is:

“All official University pages and sites must be written to be as accessible as reasonably possible to users with disabilities. As a minimum, compliance with the Web Accessibility Initiative's Priority 1 recommendations shall be required. New sites must also comply with the Web Accessibility Initiative's Priority 2 recommendations.”

The full policy statement is available at

<http://www.dur.ac.uk/its/services/web/accessibility/policy/>

1.7 The Web Accessibility Initiative

The Web Accessibility Initiative (WAI, <http://www.w3.org/WAI/>) is part of the World Wide Web Consortium (W3C), the web standards body. They have produced a set of accessibility guidelines for web content, available at <http://www.w3.org/TR/WCAG10/>

The guidelines are divided into three groups:

1.7.1 Priority 1 recommendations

Failure to meet these recommendations will stop some people accessing the content on the site.

1.7.2 Priority 2 recommendations

Failure to meet these recommendations may require some people to change their browser configuration (or in some rare cases change browser), making it difficult and occasionally impossible for them to access the content.

1.7.3 Priority 3 recommendations

Meeting these recommendations will make it easier and quicker for some people to access the site. Some of these are mainly usability improvements.

1.8 Other terms used in this Guide

Structural Markup

Structural Markup defines the *structure* of a document or provides information *about* the content. For example, the `<p>` (Paragraph) tag defines a paragraph, and the `` (Emphasis) tag shows that something is important. Attributes can also be structural. For example, the `cite` attribute of the `<blockquote>` tag shows the original source of the quotation.

Structural Markup also has a default appearance in many browsers – for example, the `<blockquote>` tag is usually displayed with an indented left margin, but it *should not* be used for this purpose. Instead, the appropriate structural tag should be used with **stylesheets** defining the indentation

Presentational Markup

Presentational Markup defines the appearance of a document in *some display media*. For example, the `<center>` tag says that the text should be displayed centred, which is clearly only appropriate to visual media, and the `` tag describes the colour, size and font used for text, and only applies to *some* visual media (for example, text-only terminals will disregard it).

Presentational Markup should not be used – instead, **stylesheets** should be used to format Structural Markup correctly.

Stylesheets

Cascading Stylesheets are attached to a document and define the presentation and layout of markup. Use of stylesheets can give greater flexibility of presentation, ensuring that content will be accessible to a wider range of browsers. A tutorial on the basic use of stylesheets is at <http://www.htmlhelp.com/reference/css/>

The use of stylesheets also allows the presentation and layout of the entire site to be controlled from a single file, making site maintenance considerably easier.

2 Finding Accessibility Problems

2.1 Examples of inaccessible pages

<http://www.dur.ac.uk/its/services/web/accessibility/inaccess/> has a demonstration of inaccessible web pages. These pages are inaccessible in most browsers. A recent version of the Lynx browser will allow you to view all of the pages normally.

Explanations of the reasons for the inaccessibility and the usual way that this inaccessibility appears are included with each of the examples.

2.2 Tools for finding accessibility problems

A list of links to online tools that can help find accessibility problems is at <http://www.dur.ac.uk/its/services/web/accessibility/tools/>

2.2.1 Web Browsers

The best tools for finding accessibility problems are web browsers. Change the browser settings such as default font size, whether pages can override specified colours and whether Java and Javascript are supported.

Use a range of different browsers. Netscape and Internet Explorer are available on the Networked PC Service, and Lynx is available on the Unix Service. If possible, try using other browsers as well.

Generally, installers for browsers, especially versions other than the latest, can be found at <http://browsers.evolt.org/>

2.2.2 Checking Tools

There are a number of automatic checking tools on the web that can check for some accessibility problems. There are a large number of accessibility problems that they cannot check for, or can only partially check for, and in some situations they will give false reports of problems.

If used with a careful interpretation of their results, then they can be very useful for identifying some problems, but one should *never* rely on them to find all problems or to guarantee accessibility - despite what some of these tools would have you believe, a page 'passing' their test only means that it passed the test, not that it is accessible.

2.3 Identifying accessibility problems

<http://www.dur.ac.uk/its/services/web/accessibility/tutorial/files/home.html> is the website for a fictional admin department. Try to identify the accessibility and usability problems on this site, using the browsers and checking tools you have available.

An example of how the site might work accessibly is at <http://www.dur.ac.uk/its/services/web/accessibility/tutorial/result/>

2.3.1 Some of the accessibility problems with the example site

- The images on the site have no alternative text (**alt**) attribute. You can spot this by disabling image loading in your browser, or using a text-only or speech-based browser.
- The university logo is also a considerably larger image than its display size (if you use Mosaic, which does not have built-in image resizing, this becomes very obvious)
- The page headings are turned white by use of the **** tag, and some text is made orange in the same way. If a Netscape 4 user has 'Always use my colours' turned on, (in Edit, Preferences, Appearance, Colours if you haven't used Netscape 4 before), then it is possible that some of this text would be invisible against the background.

- The table of contacts on the "Who's Who" page does not have proper header cells, instead just using bold, centred text. Also, some of the table cell tags are not closed, which though valid according to the HTML specification, causes display problems in Netscape 4, and also makes the page difficult to edit in Dreamweaver.
- The language used is very unclear and complex, making it very difficult to read and understand. There are also some spelling mistakes which need correcting.
- The yellow and orange text lack contrast with the blue background.
- The pages do not validate to the HTML standard, which is causing some large display errors in some browsers, especially Netscape 4.
- **Presentational markup** and attributes are used rather than **stylesheets**
- Absolute sizes are used for text and the layout table - this will become obvious if you try a narrower or wider browser window.
- No header elements are used - viewing the site in some text browsers will make this obvious as the headings will appear the same as normal text.
- Links open in a new window, which can disorientate users.
- Links do not contain any context - they are 'click here' links. They are also the same colour as the body text and not underlined, which makes them difficult to spot.
- There is no navigation mechanism or table of contents - the back button must be used to go from page to page, which causes problems if people enter the site other than at the front page.
- The pages do not have any navigation metadata. The most important and useful would be a page title using the **<title>** tag (in Dreamweaver, the Page Title box).

3 Fixing accessibility problems

3.1 Accessible text

The majority of content of most websites is text. It is therefore important that it is as accessible as possible. Text is easy for web browsers to display (although use of the proper **structural markup** can improve this) but must also be easy to read.

Studies have shown that people read 25% slower on screen than on paper, even in ideal conditions. A speech-based browser is even slower, perhaps only a quarter of the speed of reading from screen. It is therefore important that content for the web is concise and easy to scan.

Some ways to help this are:

- Place the most important information at the start of each page in the "inverted pyramid" style. Add supplementary information below this.
- Highlight important points using the **** and **** tags, or by using links.
- Use bulleted or numbered lists where appropriate, rather than a paragraph of text.

- Remove unnecessary text and 'marketese' claims. Do not make a statement if no-one would make the opposite statement. For example, a statement that you provide "reliable service" is unnecessary, since you would never claim to provide "unreliable service". However, claiming as a library to have a "wide variety of books" is not unnecessary, as another library could also positively claim to be a "specialist library in X"

Making text easier to read and removing unnecessary content can often give the impression that there is *more* content, as users can read the content more quickly, and will remember it better.

There are several articles on text accessibility and usability at <http://www.useit.com/alertbox/> The article "How Users Read on the Web" (1 October, 1997) is very relevant.

3.1.1 Example of rewriting text

The following text uses over-complicated language and is difficult to read.

"[Company name] can provide you with reliable, resilient and scaleable solutions and advice that will help your business succeed, without the stress associated with IT. Solutions that stand the test of time and evolve as your business adopts new ways of working, cost effectively."

It should be possible to at least halve the word count of this text without losing the original meaning.

An example solution

There are several areas in which improvements could be made:

1) Redundancy

The following text has had redundant claims removed, and been slightly re-ordered to retain a readable sentence structure.

"[Company name] can provide you with advice and solutions that evolve as your business adopts new ways of working that will help your business succeed, without the stress associated with IT."

2) Length

Text should be as short as possible while still retaining the original meaning. This rewrite makes the text more concise.

"[Company name] provides advice and solutions to help your business succeed without the stress associated with IT."

The rewrite cuts the word count from 30 to 16 words.

3) Simplicity of language

After the previous two stages, the language has already been simplified for ease of reading. However, one final change would be to expand the abbreviation.

"[Company name] provides advice and solutions to help your business succeed without the stress associated with Information Technology."

3.2 Accessible Graphics

Graphics cannot be displayed in every browser. Browsers such as Lynx have a text only display, handheld browsers on a mobile phone have very limited space, speech browsers cannot say a graphic. Even graphical browsers such as Netscape have an option to turn images off, for users with slow connections.

However, images in HTML can have alternative text, which will be displayed whenever the image cannot (for whatever reason) be displayed. This text may, in some browsers, also be displayed when the image is displayed, but this is not guaranteed, and you should not design expecting this to happen. Recent browsers tend *not* to do this - you should use the "**title**" attribute instead if you want text to be displayed when the user 'hovers' over an image.

The purpose of alternative text is to provide a text equivalent for the image - i.e. the text should *convey the same information as the image* when it replaces the image. Guidelines on writing good text equivalents are available from

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority1/nontext.php> or <http://ppewww.ph.gla.ac.uk/%7Eflavell/alt/alt-text.html>

Alternative text is provided for images using the "**alt**" attribute, and a link to a detailed description of the image (where alternative text is insufficient) can be provided using the "**longdesc**" attribute.

3.2.1 Using Dreamweaver to make accessible images

To add an "**alt**" attribute, select the image and edit the content of the 'Alt' box in the properties window. To add a "**longdesc**" attribute, open the tag editor (select the image, right-click, go to Edit Tag) and select the Stylesheet/Accessibility section. Enter the URL into the Long Description box.

3.2.2 Text Equivalents

It is important that alternative text is a good replacement for the image. Consider the purpose of the image when providing the text:

Images used as links

The alternative text should describe the page linked to – for example, if you use the University logo to link to the University homepage, then the alternative text "*University homepage*" or similar is appropriate.

Images of text

Images of text have associated accessibility problems as the text will not rescale with the user's font size preferences. However, if the basic size of the text is sufficiently large (14 point or above is recommended), this may not be an insurmountable problem - use of images of text for page titles, etc. is usually okay.

In this case, the alternative text should be the text displayed by the image - if the image reads "Department of Philosophy" then this should also be the alternative text.

Photographs

Consider the purpose of the photograph - if it is a semi-decorative image, consider what it is saying about the building. For example, on a college website, if you have a photograph of the front of the college, you could use alternative text in conjunction with a caption – alternative text of "*The modern college buildings:*" and a caption of "*Home to over 600 students*", which will appear in most non-graphical browsers as "*The modern college buildings: Home to over 600 students*".

You should be careful not to provide *more* information in the alternative text than you do in the photograph itself, as this is also inaccessible, although in the opposite direction to normal. If you are providing more information in the alternative text, consider moving some of it to a caption.

The alternative text "*Photograph of college buildings*" is a *description* of the photograph, not an *equivalent* for it, and so would not be suitable.

Diagrams and Graphs

Diagrams are often used to convey very complex ideas. Therefore, replacing the content of the diagram or graph with alternative text will always be inadequate. Try to summarise the diagram or graph with alternative text, and then either provide a link to a detailed description of the diagram or graph (using both a normal link and the "**longdesc**" attribute), or explain the diagram or graph in the text of the document as an extended (possibly several paragraphs) caption.

This will not only help people who cannot view the image, but also people who can view the image but are unclear about the meaning.

Decorative Images

Purely decorative images that serve no other purpose should be avoided where possible. However, if used, they should have alternative text of a single space " ". Leaving out the "**alt**" attribute will cause many browsers to display the image as "[IMAGE]" or similar, which is unhelpful.

3.2.3 Examples of 'text equivalents'

There is a series of images at

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority1/examples/nontext.php>

Example text equivalents using the guidelines above are available on the same page.

3.3 Accessible Tables

There is a lot of data best expressed as a table. To make tables more accessible, they should be marked up with headers (`<th>` tags). The visual appearance of a header is usually (though this can be changed) bold and centred text. However, the header tag also has meaning, which improves accessibility and document searching, and also allows interesting possibilities for automatic processing.

Depending on the complexity of your tables, the specification of headers can be more or less complex. Dreamweaver, unfortunately, does not have an easy way to fully specify headers, though you can of course still edit the code directly.

Tables can also be used for page layout. If possible, try to avoid this, as it can cause some accessibility problems. However, the accessibility problems are small compared to the problems of inaccessible text, images or data tables.

Headers must be assigned to data cells. There are two major ways to do this:

The Scope attribute

This attribute can be used to give each header cell a set of data cells to apply to. There are four possible values for **scope**:

- **row**: all cells directly to the right of this one.
- **col**: all cells directly below this one.
- **colgroup**: all cells below this one in the same column group (defined by the `<col>` and `<colgroup>` elements).
- **rowgroup**: all cells to the right of this one in the same row group (defined by the `<tbody>` elements)

Where it is not clear whether a cell is header or data, use a normal data cell, but give it a scope.

The "Headers" attribute

Where the "scope" attribute takes a header cell and defines the data cells associated with it, the "headers" attribute defines the header cells associated with each data cell. Each data cell has a space-separated list of header cell "id" attributes in its "headers" attribute. Header cells simply have unique id attributes.

This method is more flexible, and more widely supported by browsers, but can be considerably more work to implement.

Where it is not clear whether a cell is header of data, use a normal data cell, give it an "id" attribute, and a "headers" attribute that points towards its header cells, and then add its "id" attribute to the "headers" lists for other appropriate data cells.

3.3.1 Using Dreamweaver to add header information

To change a data cell (<td>) in a table to a header cell (<th>), select the cell, and then check the Header checkbox in the cell properties window. You can set the 'scope' of a cell by selecting it, right-clicking, and going to Edit Tag. There is no automatic way to set the 'headers' attribute or to create row or column groups - you must do this by editing the code directly.

Therefore, in Dreamweaver, use of the 'scope' method for assigning headers to data cells is much easier, provided that the table does not require headers to apply to row or column groups.

3.3.2 Example of adding headers to tables.

There are two data tables on this page. One has a single set of headers, the other a more complex set of headers.

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority1/examples/tables.php>

Example solutions are available on the same page, using both 'headers' and 'scope' methods. Looking at the page's HTML source or viewing it in Dreamweaver should make these guidelines clearer.

Example Table

<code><th scope="col" id="th1">Heading 1</th></code>	<code><th scope="col" id="th2">Heading 2</th></code>
<code><td headers="th1">Data</td></code>	<code><td headers="th2">Data</td></code>
<code><td headers="th1">Data</td></code>	<code><td headers="th2">Data</td></code>
<code><td headers="th1">Data</td></code>	<code><td headers="th2">Data</td></code>

3.4 Accessible use of colour

Colours can never be relied upon to provide information.

Non-visual browsers cannot display colour, and most text only browsers do not display author-specified colours. Almost all graphical browsers have an option to use user-specified colours and so ignore author-specified colours entirely.

About 4% of the population has some form of colour blindness, usually red-green, although blue-yellow also exists.

Therefore, any information given by colour must be given by other methods as well. The method that is appropriate will depend on the situation.

Also, use of the **** tag to set colours can cause large display problems in some older browsers such as Netscape 4 if author colours are ignored, as a bug in these browsers means that only the author-specified *page* colours are ignored. The **** tags will still be used, which can give text disappearing into the background.

<http://www.mcsr.olemiss.edu/~mudws/font.html> gives a description of some of the problems with ****. While the majority of this was written in 1996, the problems are still present.

3.4.1 Partial specification of colour

Only specifying a background colour or a text colour can cause large problems. You should specify both or neither.

Consider the situation where the page background colour is specified as white, but the text colour is unspecified, and the user's default colours are black background, white text. The page background colour will be white, the text colour, not specified in the page, will fall back to the user's default – also white.

To avoid these problems, always specify a background colour and a text colour or leave both unspecified. If you are using stylesheets (recommended), use both or neither of the **background** and **color** properties on each style specification. If you are using the **<body>** tag to specify colour, make sure you specify all five of the **bgcolor**, **text**, **link**, **vlink** and **alink** properties.

Since the **** tag only sets foreground colour, using it can easily cause this sort of problem.

3.4.2 Specification of colour in Dreamweaver

Dreamweaver uses the **** tag by default to set font sizes, colours and typefaces. You should instead use Cascading Style Sheets (CSS) to set these.

<http://www.htmlhelp.com/reference/css/> gives a guide to using basic CSS. To link a stylesheet to a document using Dreamweaver, click the 'link' icon in the head section, or use the Insert->Head Tags->Link menu option, giving a 'rel' of "**stylesheet**".

Alternatively, specify the colours using the **** tag as normal, and then run "**tidy -c**" on the output (see *Infosheet 163*).

3.4.3 Examples

There is an example of an inaccessible use of colour at:

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority1/examples/apppearance.php>, a page listing support levels for software.

Consider alternative ways of presenting the information on that page that does not *rely* on colour to do so. Colour may still, of course, be used to present the information, provided that an alternative way also exists.

There are some example alternative presentations on the same page. The alternative presentations retain the colour as well, but need not do so.

Providing information with each item

Each item has the support level listed after it in brackets. The support level remains in colour but is written out explicitly.

Grouping items using headings

The items are grouped using the `<h4>` Heading tag. The heading lines remain in colour. Heading level 4 was used as it was the next logical level of headings on this page, and obviously this will differ from page to page – see the section on Structural Markup below.

Specifying the information in a table

The information is specified in a table with table headers specified. Since tables are used, there are additional accessibility considerations; however this may be the most appropriate way to display the support levels in some circumstances.

3.5 Scripts, Java applets, Flash and similar technologies.

All of these technologies are inaccessible by default, like images. Again like images, there are ways to provide equivalents. It must be possible to use the site with one or more of these technologies disabled.

Clearly, a Java applet that does real time 3D rotation cannot be duplicated exactly in text, but pictures of the object from different angles, with appropriate descriptions for people without images (and some people with), could provide an adequate alternative.

Since making these technologies accessible is somewhat more difficult than making images accessible, try not to use them unless it is the best way of displaying the content.

Client-side (i.e. on the user's browser) scripting such as Javascript, and the equivalent functionality from Java applets or Flash animations can be used to provide interactivity. First, however, you should consider if it is possible to provide the same using Server-side (i.e. on the web server) scripting such as PHP to perform the same effect, as this will work in all browsers.

You should avoid using scripts, applets or flash for navigation or normal text content, and must provide a good alternative with normal links if you do. Remember that as well as being inaccessible to users, script, applet and flash based navigation or text content without an alternative is invisible to a search engine, which will considerably harm search engine results.

In summary, use these *only* when normal text and images cannot convey the content, or where server-side scripts cannot provide the required level of interactivity.

3.6 Frames

Frames can be made accessible, but it is very difficult to do so. The accessibility guide

(<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority1/frames.php>)

has more information if you need to use frames. In most cases (except on very small sites), it is easier to remove the frames and replace them with a table or stylesheet controlled layout than it is to make the frames accessible.

For general pages, frames are not necessary, and have a number of accessibility problems including:

- Failing in non-graphical browsers unless large efforts are made to provide alternatives using the **<noframes>** tag (in Dreamweaver, select “Modify, Frameset, Edit NoFrames Content” from the menu). The common alternative “Sorry, your browser does not support frames” is *certainly not accessible*.
- The majority of browsers cannot bookmark frames. Therefore, users will be unable to bookmark anything other than the front page of your site without considerable extra effort, which will reduce its usability to them.
- Most search engines deal with frames very badly, and rely on the **<noframes>** content.

There are occasional circumstances where the use of frames could be appropriate – for example, comparing two documents. However, an accessible alternative that works without frames must also be provided.

4 Other accessibility issues

The improvements in accessibility in the previous section should allow users to view the page in any browser, but it may still be difficult for them to do so, or require them to change their browser settings. To make a page reasonably *easy* to read for all users requires a further set of accessibility improvements.

A full description is available at

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority2/> but the most important points are outlined below.

4.1 Structural Markup

The largest accessibility improvement can be achieved through the correct use of HTML tags to describe document structure. For example, while the **<blockquote>** tag is displayed indented in some browsers, it should only be used to mark *quotations*. The misuse of HTML markup to achieve presentational effects can cause large accessibility problems. Cascading

Style Sheets can be used to create presentational effects such as text indentation, colours, and size, as well as more complex effects such as borders and entire graphical layouts.

A major improvement in this category is the correct use of heading tags (<h1> to <h6> - Text->Paragraph Format->Heading 1 to 6 in Dreamweaver). Heading tags should be set out in the logical order, with a single <h1> (Heading 1) at the top of the content (navigation and template elements may go above it, of course), with <h2>s (Heading 2) nested below it, and <h3>s (Heading 3) below them, in a hierarchical structure. Do not break the ordering of heading tags simply to achieve a formatting effect, as the visual effect can be duplicated by stylesheets without reducing accessibility.

Where a formatting effect is desired, but the content has no appropriate HTML element, <div> (for block elements) or (for inline text) with a stylesheet applied can be used. These can also be used un-styled to sensibly split up content for which there is no markup - for example, an address could be marked up as:

```
<div class="address">
  <div>Old Shire Hall</div>
  <div>Durham</div>
  <div>DH1 3HP</div>
</div>
```

If there is an appropriate HTML element, however, it should be used, as this will help display of the page on browsers without stylesheet support.

4.2 Validation

Creating code that validates to a recognised document type is helpful for display between browsers. Invalid code can sometimes trigger bugs in browsers that can make content unreadable in some browsers, whereas other browsers will guess the correct interpretation. One well-known issue is for sites with a missing </table> tag to be invisible in Netscape 4, which refuses to guess a table layout until it is *certain* that the table has ended.

Valid code can also become unreadable in browsers, of course, but it reduces the potential problems. Dreamweaver will usually produce valid code or almost valid code, and use of "tidy" can usually fix any problems.

It is considered polite to ensure that your code validates before asking for help fixing display problems. An exception is of course made if you are asking for help to make code valid...

HTML validators are at <http://validator.w3.org/> and <http://www.htmlhelp.com/tools/validator>. A CSS checker, which also gives some warnings about code which may cause problems, is at <http://jigsaw.w3.org/css-validator/>.

4.3 Navigation

A clear and consistent navigation structure can make it much easier to move about the site. There are a number of techniques that can improve

navigation accessibility and usability, not all of which are useful on every site (for example, site maps are usually pointless on small sites)

- Place navigation bars in a consistent location on each page.
- Make clear the location of the page within the navigation hierarchy, for example with a so-called breadcrumb trail - e.g. "You are at: University of Durham > Information Technology Service > Web". This is only useful on sites with a strong hierarchical structure.
- Provide a link back to the site homepage on each page. Within the University, you should also link to the University homepage on official sites.
- Make sure link texts make sense when read out of context. Some browsers allow the user to look at (or hear) a list of links on the page for quick navigation, so it must be possible to guess the link destination without reading the surrounding text. Links such as "here" or "click here" are especially unhelpful and should never be used. Since some search engines use link text to help classify pages, link text that does not work out of context can also harm search engine results – do a search at <http://www.google.com/> or for "click here".
- Link within paragraphs to related content where appropriate.
- Include a site map and/or search facility. This is especially useful on larger sites.

4.4 Other areas to consider

More information on these areas is available on the accessibility guidelines site at

<http://www.dur.ac.uk/its/services/web/accessibility/guidelines/priority2/>

4.4.1 Tables for layout

Under the structural markup guidelines above, you should not use tables for layout. However, it takes some work to convert to a stylesheet-based layout, so until this is done, ensure that the table makes sense when linearised (i.e. when the cell contents are listed left-to-right top-to-bottom) - you can check table linearisation using Lynx on the Unix Service, or the Web Page Backward Compatibility Viewer at <http://www.delorie.com/web/>.

4.4.2 Forms for user input

Form controls should be associated (by position and preferably also by the `<label>` tag) with sensible labels, and hints and instructions should be provided where appropriate.

4.4.3 Colour contrast

Colour contrast should be good, and take into account colour-blindness (red-green is the most common, but blue-yellow is known). This is more important for images than text, as text colours can be changed easily by the user. Combining colour contrast with brightness contrast is a useful way to do this.

A simple test is to print the page to a black and white printer, and check that it is still easily readable. You can also use the **VisCheck** software at <http://vischeck.com/vischeck/>

5 Useful tools

A list of useful tools to help you check accessibility is available at <http://www.dur.ac.uk/its/services/web/accessibility/tools/> Links to the tools mentioned in this section are available from there or on *Infosheet 164*.

5.1 Browsers

The most useful checking tool is a good range of web browsers. To get a full range, you ideally need 3 computers - a Mac, a Windows PC, and a Linux/Unix PC (or with Mac OS X, Linux/Unix software installed on the Mac).

If this isn't possible, the majority of browser types are available for Windows in one form or another. A KHTML-based browser (such as Konqueror on Linux or Safari on the Mac) for Windows has not yet been released, but development is in progress.

Make sure you have at least one text browser, at least one graphical browser, and at least one screen reader or speech browser, if at all possible.

The principle of an accessible website is that it should be possible to read the content in any browser, without the user having to change settings.

A range of browsers can be downloaded from <http://browsers.evolt.org/>

5.2 Validators

HTML and CSS (stylesheets) both have formal specifications. There are several automatic tools to check a site against these specifications, and you should use these. The tools list has links to HTML validators and CSS checkers.

Validation and accessibility are separate - it is possible for a valid page to be inaccessible, and possible for an invalid page to be accessible (depending on the nature of the invalidity). However, for more reliable display of content, especially in browsers that you do not have available to test with, valid files are best. If there is a layout problem with a page in some browsers, ensuring the code is valid can often fix this.

5.3 Accessibility Checking Tools

Tools like the W3C Accessibility Valet, Bobby or A-prompt can be very useful for quickly checking for accessibility problems. However, they can not find all problems, and can give both false negatives and false positives. You should never rely on them, but you should use them where appropriate as useful tools for picking up some more common mistakes.

For example, when checking that there is an appropriate 'text equivalent' for an image, Bobby will check for the presence of the alt attribute. If it is

not there, then it will report an error. If it is there, but has entirely *inappropriate* content, then Bobby will *incorrectly* report no error.

The URLs <http://www.dur.ac.uk/Department/index.htm> and <http://www.dur.ac.uk/Department/> often point to the same page. However, when Bobby checks that links with the same link text go to the same page, it will flag these as different pages (since it cannot know that the web server is set up to make them the same), and give an error where none exists. Also, the requirement for all links with the same link text to go to the same location, while a useful guideline, is in a few situations better to ignore – however Bobby assumes that it must be met in all situations.

These tools can be useful for picking up basic accessibility problems, but almost *all* accessibility problems require some degree of manual checking.

5.4 Other tools

5.4.1 Specialist accessibility checking assistants

Tools like **Vischeck** or the **Backwards Compatibility Viewer** do not claim to check accessibility. However, they can provide useful information, which if interpreted correctly, can allow you to identify and correct accessibility problems.

5.4.2 Link checkers

Broken links are a usability rather than accessibility problem, but are definitely worth finding and correcting. On a large site, it is easier to use an automatic link checker such as **Xenu** - read *Infosheet 161* for more details. It is installed on the Networked PC Service.

5.4.3 Tidy

tidy can convert invalid HTML into valid HTML or XHTML. It also has a range of options to deal with HTML produced by Word, work with server-side scripts, convert **** tags into stylesheet code, and several other features. Read *Infosheet 163* for more details. It is installed on the Unix Service.

6 Summary

- Check your sites carefully
- Fix problems under the Priority 1 Recommendations quickly
- Use automatic checking tools, but be careful to interpret the results properly
- Use a wide range of browsers and browser settings
- Think carefully about your text content and about text equivalents for your images.