

Keeping your website accessible

The intention of this article is to help you to keep your website accessible.

All of the 'in practice' examples are based on the use of Wordpress; which is currently the world's most popular content management system (CMS). However, the techniques will apply equally to other content management systems, as they tend to work in similar ways, e.g. they all use WYSIWYG (what you see is what you get) editors, image and upload and management features and provide a plugin based architecture.

They also tend to display similar accessibility weaknesses (e.g. plugins that generate lots of 'read more' links).

The techniques outlined are not complicated. Taken one at a time they are actually quite simple.

So take a deep breath and let's get started.

Some notes about your basic website design and site organisation

Assumptions apply

This guide assumes that the websites your website has been designed - at the outset - to be accessible. It also assumes that the content management system used doesn't itself introduce access issues or make it difficult for you to keep your site accessible.

Ensure your design is accessible

Your basic website templates should be accessible and built with good practice in mind (e.g. WCAG 1 or 2, BS 8878:2010) ; adhering to these guidelines is your website developers job - but you can help ensure they follow these guidelines by specifying them in your tender document. Some useful tips on tendering can be found in this document: <http://www.jimbyrne.co.uk/fileuploads/Writingtenderdocuments.pdf>).

Ensure markup and accessibility standards were used

Pages should have been built with error free code and conform with the accessibility standards set out by the World Wide Web Consortium (i.e. it should be valid and standards based). However, it is only practical and realistic to assume that perfection cannot be achieved; the aim to ensure as much as you can that disabled people are not being discriminated against.

It helps if your content is well organised

Sites content should be well organised and have consistent, easy to understand navigation. The main navigation elements should be in the same place on each page, for example, if the main section navigation is set out horizontally across the top of the page don't have some pages or sections where it is set up vertically. If sub-navigation is always on the right of the page - don't have it on the left on some pages. Variety is the spice of life, but not on your website; when it comes to navigation, convention is your friend.

Consistent navigation helps everyone. It particularly helps people with learning difficulties, people who are blind or have a visual impairment and people with motor impairments.

Pages themselves should also be well organised - with headings and sub-headings to show how sections relates to each other. Think of how the front page of a newspaper is organised, i.e. huge banner headlines, sub-headings, stories illustrated with images. Web pages should following this simple well-tested pattern for displaying content in a logical manner.

Use Skip links

Skip links should be added to page templates so that non-visual users can jump to important parts of the page without having to tab through lots of links. Skipping directly to the content area of a page - or to the search form or to sub-sections navigation help screen reader users navigate around your pages.

Flexibility is the key

An accessible website is one where the presentation of the site content can be changed by the end user; i.e. the user can change the size of the text, the contrast colours, even the layout (e.g. by turning off the default style sheet or substituting their own). If someone is finding the text too small to read comfortably they should be able to make the text bigger themselves; if the contrast between background colour and text colour is too great (or too low) they should be able to choose different colours.

Most web browser provide the ability yo make these changes via their 'preferences' screens. The key is to ensure that the web designer has not designed the site is such as way as to override that flexibility (e.g. using fixed point sizes for text or old-style markup to set presentation elements such as text colours within the page).

Many people - including many website designers - are not aware that browsers such as Firefox, Safari, Chrome and Internet Explorer allow site

visitors to change these presentation elements. Now that you do; you can use those preferences yourself to check that the presentation of your websites can be changed by visitors to your site.

Techniques for keeping your website accessible

Add labels to non-text elements

Images (and other non-text elements) should have text alternatives. This is to ensure that those who can't access visual information directly (e.g. a blind person) do not miss out on that information.

I'm referring to Images in my examples as they are the easiest to understand - however the same principle applies to other non-text media such as video and audio. An audio transcript is an example of a text alternative.

Images should have text labels that describe their function. When adding a text label, you need to think, 'is the content of this image important - or is it the function of the image that is important? If it is an image of a product - then clearly it is the content of the image that is important, i.e. it would be picture of the product; so your text label would describe what the product is. If it is an image of person, you give the persons name in the text label.

If it is an image of an outdoor scene you could describe the scene:



```

```

But what if the image is a button that the visitor is expected to press? You would not describe the look of the button; you would add a label would reflect the function of the button; so the text label might say 'Submit' if it was for submitting a form or it might be the link text that takes the visitor to a different page on the site, e.g. it might say 'About us' if it was a link to the about us page on the site.

Or it might be the submit button for a search form:

A rectangular button with a thin black border and a white background. The word "SEARCH" is written in bold, black, uppercase letters in the center of the button.

```
<input type="image" src="http://www.somesite.com/images/search/sa.gif" alt="Search" width="69" height="31" />
```

Decorative images

Images that are purely decorative should have empty labels - i.e. there should be an alt attribute but that alt attribute should not contain anything, i.e. alt="".

Background images are not accessible

Don't use background images for important content (e.g. logos, navigation bars or images that contain information such as charts) as it is not possible to attach labels to background images. If you use background images for your navigation then it will be impossible for some users to navigate your site (because they will not know where each link will take them as the alternative text won't be available to them).

Leave the alt attribute empty?

It's not always a good idea to add text labels to images, sometimes you are better just leaving the alt attribute empty. If for example, the image label says exactly the same as a heading next to it, then it would be better to leave the alt attribute empty; otherwise a screen reader would just read out the same text twice; which is a waste of time.

Always consider whether adding a text label helps or hinders the visitors understanding of the content; if it's not helping leave the alt attribute empty. You can get a quick sense of how useful or not your text labels are by viewing your web page with images turned off.

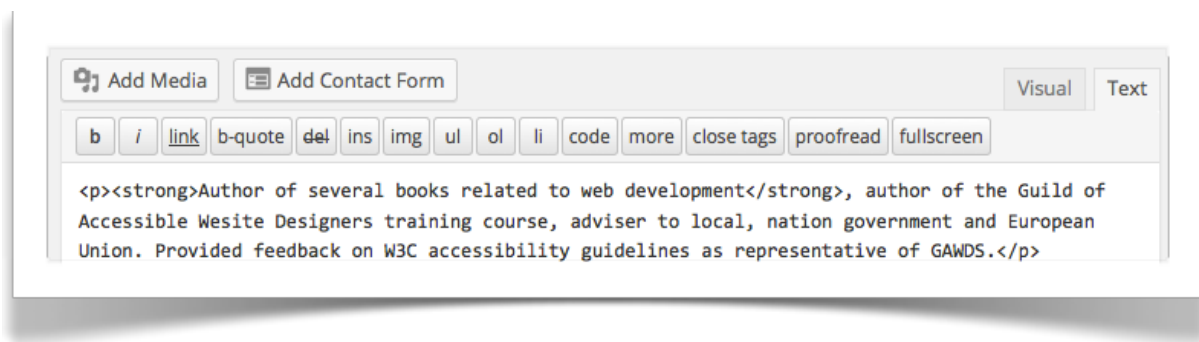
In Practice: adding alternative text to images

How to add a label to an image

Not every content management system makes it easy to add labels, so it is useful to know a little bit about how to create (or fix) the code for labels yourself. This is not difficult to understand - here is an example.

Most content management systems have WYSWYG (what you see is what you get) editors that have SOURCE options to allow you to edit the code yourself.

In the case of Wordpress you can view the HTML source by clicking the 'Text' tab located on the top right of the editing toolbar.



Code without a label:

```

```

Code with a label:

```

```

Code with an empty label

```

```

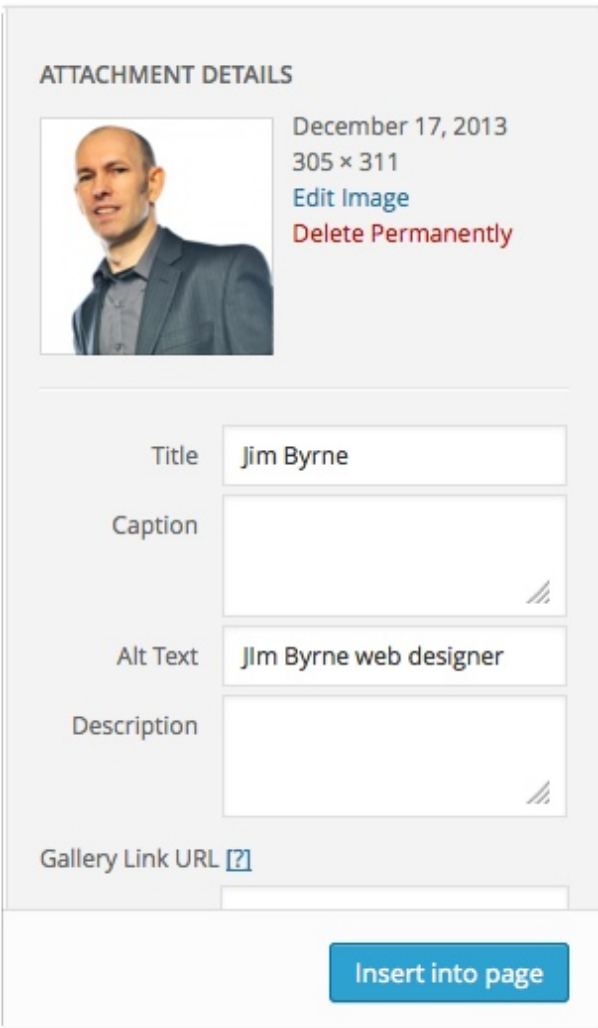

The reason you want to add these labels is so that a person using a screen reader can have the text label read out; i.e. in the above example they will hear, 'Jim Byrne Web Designer'.

Here is how it's done in Wordpress

Wordpress makes it easy to add a label to an image before it is placed in the page.

When you click the 'Add Media' button when editing a page in Wordpress - you get the opportunity to upload a new image or input an existing image.

Any images you upload will appear in the image grid in the main content area of of the page. When you choose an image from that grid you will see a form on the right-hand side.



The screenshot shows the 'ATTACHMENT DETAILS' form in WordPress. At the top left is a thumbnail of a man in a suit. To its right, the date 'December 17, 2013' and dimensions '305 x 311' are displayed. Below these are two links: 'Edit Image' in blue and 'Delete Permanently' in red. The form contains several input fields: 'Title' with the value 'Jim Byrne', 'Caption' (empty), 'Alt Text' with the value 'Jim Byrne web designer', and 'Description' (empty). At the bottom, there is a 'Gallery Link URL' field with a help icon and an 'Insert into page' button.

On that form you would fill in the value of the 'Alt Text' field; this will add an alt attribute (i.e. an 'alternative' text label) to your image when you insert it into the page.

You will be remembering that the label used should reflect the function of the image; e.g. if you are inserting an image that acts as a link (or as a button) - the label will tell the user what will happen if the button or link is clicked, i.e. a search button will say 'Search' - a link to another page will tell you the name of the target page.

If the image shows a picture; a description of the picture is appropriate (e.g. A Scottish mountain scene) - if it's a person - the persons name and any other relevant information should be provided (e.g. Chief Operating Officer Jim Byrne).

Don't confuse the alt attribute with the title attribute

You will notice that when you hover over images on some web pages you can see a 'tooltip' popping up, i.e. a text description of the image that hovers over the page. This is because the title attribute has been added to the image; it does not however mean that the image label is accessible.

Although you will not see the tooltip after adding your alt attribute (unless you are using Internet Explorer) - that does not mean that it is not working properly; the alt attribute and the title attribute have different jobs to do; and in terms of accessible the alt attribute is the one to use.

Keyboard users for example will never see the tooltip (as the tooltip is only shown during mouse over) and screen readers are not set by default to use it.

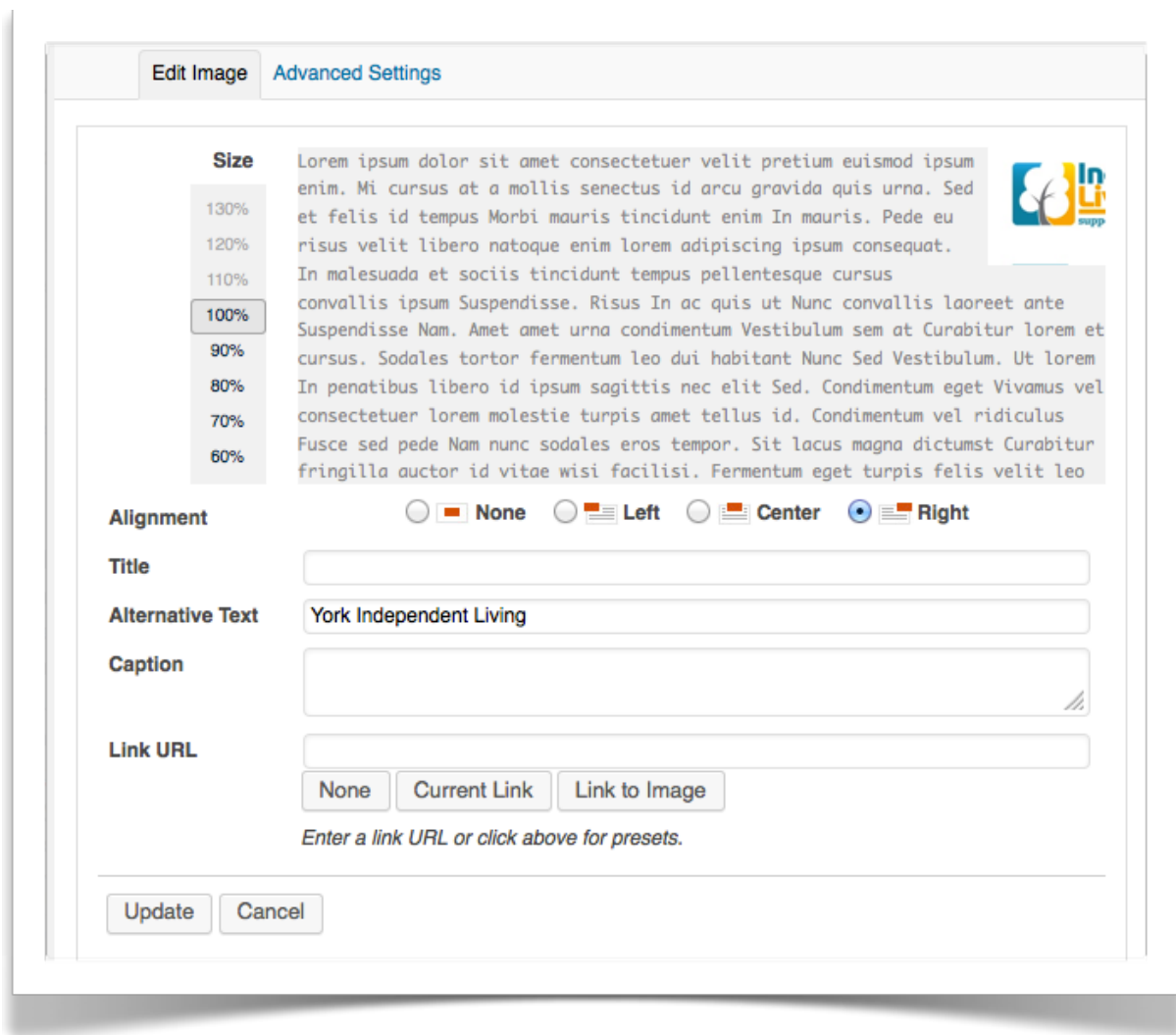
Don't use the title attribute on images

The title attribute is very useful and a requirement for some elements such as the *acronym* or *abbr* where it provides useful contextual information. However, it is not recommended that you use it on images as it can cause problems for some users, e.g. it can be an unwanted distraction for people

with Dyslexia and it can block access to the Alt attribute for some screen magnifications users.

In practice this means that when you edit an image in your CMS you should not fill in the value for the title attribute.

In Wordpress it only gives you the option to add the title attribute after you have inserted the image. You get this option when you choose to edit your image by clicking it in the edit page and choosing 'Edit Image' (a link that is shown top left of the image when you hover over it).



Confusingly, the Title field you will see when you are in the 'Insert media' area is not the same as the title field shown for the image when you editing Jim Byrne Accessible Website Design Glasgow <http://www.jimbyrne.co.uk>

it after having inserted it into a page - even though that field is also called Title.

For a more detailed article and additional examples related to adding labels to images with examples see the page at <http://www.jimbyrne.co.uk/show.php?contentid=216>

Help your visitors navigate between and within pages

It's not just your site that needs to be well organised - page content should also be well structured and easy to understand: page sections should have headings, sub-sections should have sub-headings and your content should reflect the organisation of those headings and sub-headings.

Page structure can, and should, be represented visually - but it also needs to be logically 'marked up' with appropriate html elements - to ensure that this it is also accessible to people who cannot see the page.

A web page is a 'structured' document; and the markup (i.e. the HTML tags) should reflect that. This is important both for machines reading your pages (e.g. Google) and people.

Newspapers have headings and sub-headings to organise content and to tell readers what is most important and what is related but less important; headings and subheadings on a web page have the same job to do.

For a more detailed explanation of what is meant by 'structured markup' and standards based HTML coding visit the pages at <http://www.jimbyrne.co.uk/show.php?contentid=215> and <http://www.jimbyrne.co.uk/show.php?contentid=217>

It is possible make pages look as if they have appropriate headings by visually making headings bolder and bigger (i.e. by using the bold button on the WYSWYG toolbar and increasing the font size) - but that in itself is not an accessible approach. Headings still need to be marked up as headings by using heading tags, paragraphs need to be marked up as paragraphs by using paragraph tags, and lists marked up as lists by using list tags.

By 'marked up' I just mean that the type of content has been labeled as a heading or a paragraph or a quote or a list or a table...., e.g. in the following example the heading has been marked up using the h1 element:

```
<h1>About Us</h1>
```

The h1 is saying to the web browser, *'this is the main heading on the page please display it appropriately'*. It is also saying to a screen reader, *'voice this in a way that indicates it is the main heading for this page'*. And it is saying to Google, *'this is the main subject of this web page please index it as such in your searchable database'*.

'Marking up' the elements of a page ensures that the organisation of that content is accessible to people using screen readers; the screen reading software will know which bits of the texts are heading and what the relationship is between those headings (i.e. main heading, sub-heading). Marking up text using appropriate headings makes that possible (i.e. H1, h2, h3 and so on).

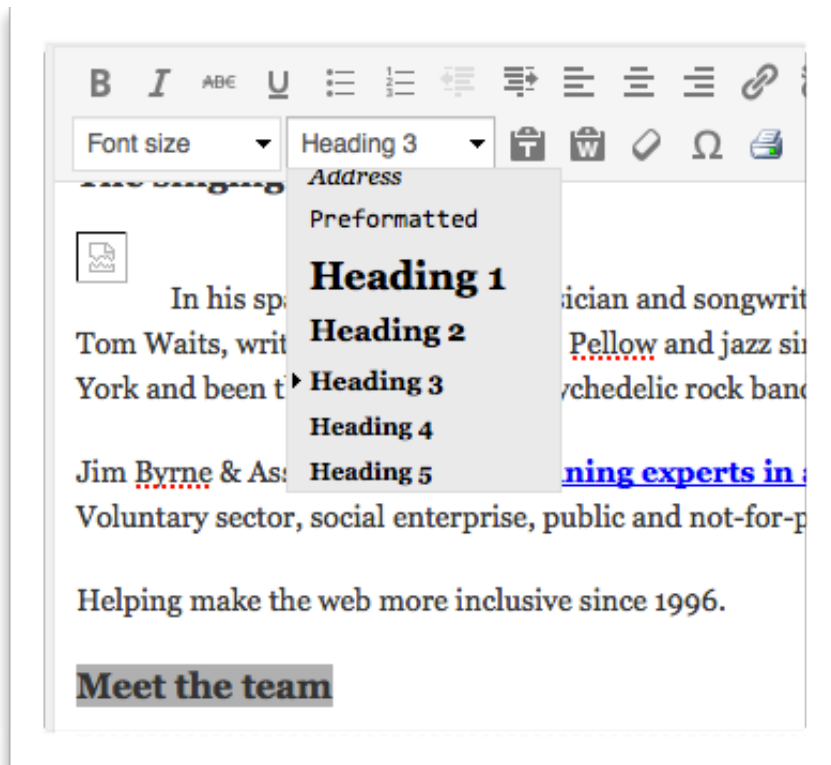
For that to work - the website text editor has to be set up correctly - so that when you make text into a heading - it is not just made bold - but it actually has the appropriate HTML tag added.

Markup your content correctly

What the above explanation means in practice is that you need to learn how to use the WYSWYG toolbar works in your content management system to markup your page content appropriately.

In practice: how to create headings in Wordpress

When creating headings you need to highlight your text and then use the format pull-down menu to format your heading, i.e. as a Heading 1 or a Heading 2 or a Heading 3 or whatever is the correct level for that particular heading in the page.



When you want to create a list you highlight your list content and click the list button on your WYSIWG toolbar and if you want to create a paragraph you highlight your text and choose 'paragraph' from the formatting menu.

Paying attention to this aspect of page content creation ensures that you are adding a logical structure to your pages; which will be very helpful to people using screen readers (among others).

The importance of descriptive link text

Ensure that the text used for links adequately describes the page or content you are linking to. Link text should be short and descriptive and still make sense when read out of context.

There should be no '*read more*' type links on your pages. This can be a particular issue for content management systems using plugins that generate their own links; for example, many news or events plugins show lists news items with 'read more' links appended to every item.

Not only does this mean these links do not make sense when read out of context - but it also means that each link is not unique. Both of these are major accessibility issues. Screen reading software has the facility to summaries all the links on a page for a user; if they all say, 'read more' - that's worse than useless - it's inaccessible and it's also annoying.

Generally speaking, those 'read more' links are not easy to fix (unless you can hack code) - so what you need in this instance is take advantage of any accessibility plugins that are available for your content management system. In the case of Wordpress there is a plugin called 'WP Accessibility' that can help solve this problem.

The WP Accessibility plugin will add the title of a post to automatically generated read more links (Download the plugin from <http://www.wordpress.org/plugins/wp-accessibility/>)

Underline links in page content

Link in the main content area of pages should be underlined; this is a well known convention - it tells a visitor that it is a link and not just another bit of text; underlined link text increases usability as it makes it quicker for people to find them.

You won't need to do anything for links to be underlined; that is the default style; so if links are not underlined it means that your designers has overridden the default link style.

It is also a good idea to to indicate change the contrast or colour of a link when a user focuses on it - either by tabbing to it or using their mouse to hover over it. This will help some of your visitors to navigate through your page - they will know what link is being highlighted at any particular point in time.

Again, the WP Accessibility plugin can help with focus; it will 'add an outline to the keyboard focus state for focusable elements.'. A very useful plugin.

Don't use web addresses (URLs) as link text

Using a URL as link text presents accessibility and usability problems - in particular - to people using screen readers. Characters in web addresses can include all sorts of funny characters that are difficult to understand. URLs can be very long and make no logical sense; particularly as many pages are now served directly from databases.

Links should stand out

As well as underling links they should also be a different colour from the surrounding text. WCAG 2 stipulates that if you fail to underline your links you must make the stand out using contrast: a 3:1 contrast ratio. This is to ensure that people with poor vision or color deficiency can still identify links.

Understand colour contrast and accessibility

Colour - in terms of accessibility - is one of the areas I find hardest to understand; I can read a sentence like, 'avoid using colors of similar lightness adjacent to one another, even if they differ in saturation or hue.' (from http://www.lighthouse.org/color_contrast.htm) and be as confused after I've read it as I was before.

I guess that is because; not having done a course on colour theory, I'm thrown by the jargon. In order to understand this we need to define the words hue, lightness and saturation, and, having figured out what they mean - try to understand the above sentence from lighthouse.org. Bear in mind that with these definitions I am simplifying as much as I can.

Hue:

This is the easy one - just substitute the word 'colour' for the word hue and you have the meaning.

Lightness:

How much light does the colour reflect: black doesn't reflect much, white reflects lots. Colours thus appear light or dark; how light or dark they are - tells us their 'lightness'.

Saturation:

The purity of the colour - saturated colours contain no white, grey, black or complementary colours.

Ok - so now I'll put the quote from the Lighthouse website into words I understand: "Even when using different colours next to one another (e.g. text and a background colour), if they are similarly light or similarly dark there will still be accessibility issues for some users."

Perhaps this 'colour business' is not as impenetrable as I thought. :-)

A quick way to check colour contrast

Take a screen shot of one of your web pages, open it up in an image editing program (e.g., photoshop) and desaturate the image to remove all colour. You will end up with a greyscale image. This will make it easier to see the contrast between different colours and/or between overlapping images.

Check colour contrast using freely available tools

Colour blindness

People with colour blindness and other visual impairments can have issues with the contrast between certain colors. There will be people who can't see the colours on a page at all, e.g. they might be viewing your web page on a greyscale monitor or be listening to your page content rather than viewing it. This means you should not rely on colour alone to provide important information

For example, when creating a web form don't write, 'the fields with a red dot next to them are compulsory, those with a green dot are optional.' This statement will be of no use to people who are colour blind, or those using grey-scale monitors, or for that matter, those using screen readers.

Here are the main colour combinations to avoid for people who are colour blind:

- Red/green combinations (memory aid: red berries against green leaves on a tree)
- Blue yellow combinations (memory aid: yellow daffodils against a blue sky)

'The Institute for Dynamic Educational Advancement (IDEA) and Brandeis University' provide some useful information about colour blindness at <http://webexhibits.org/causesofcolor/2.html>.

There are various tools you can use to test whether the contrast is good or not.

Ensure your forms are accessible

Forms are an extremely important part of the web; simple contact forms, booking forms, shopping basket forms, registration forms and so on. It pays to ensure that they are accessible and easy to use.

The most common access issues include;

- Labels not close to, or not clearly associated with their input fields.
- Confusing layout and poor organisation of forms.
- Inaccessible submit buttons.
- Lack of keyboard access.
- Poorly implemented Javascript.

Associate form fields explicitly with their labels

When encountering a form on a web page it is easy for sighted users to work out the expected response for each form element; it is just a matter of reading the label next to the field.

However, it is not always quite so simple for someone using a screen reader. If labels are not physically close or clearly associated with the form element, for some users, it may not be clear which labels are associated with which field (particularly if the page author has used tables to layout the form).

Use the following technique to explicitly relate labels with their form elements; each label can be identified with a name, and then associated with the appropriate control using the 'id' attribute, e.g.,

```
<label for="name">Name</label>
```

```
<input name="Name" type="text" id="name">
```

The value of the 'for' attribute can be any text you choose - as long the appropriate id attribute has the same text.

Using this technique also provides a bonus for people with impaired motor skills because it increases the 'target area' when assigning focus to the field, i.e. clicking the label or the field itself will focus the cursor in the text field (or check the box if it is a radio button).

Group related form fields

It is a good idea to divide forms into logical units using the fieldset and legend elements. Here's an example of grouping two radio button fields for choosing between two colours.

```
<fieldset>  
  
  <legend>Do you like blue or red?</legend>  
  
  <input type = "radio" name = "blue" value = "blue"  
  checked id="blue"> <label for="blue">Blue </label>  
  
  <input type = "radio" name = "red" value = "Red"  
  id="red"> <label for="red">Red</label>  
  
</fieldset>
```

The above code will produce the following form choices:

A screenshot of a web form. The form is enclosed in a light gray border. At the top, the text "Do you like blue or red?" is displayed in a bold, black font. Below this text, there are two radio button options: "Blue" and "Red". The "Blue" option is selected, indicated by a blue dot in the center of the radio button. The "Red" option is unselected, indicated by a gray dot in the center of the radio button.

Don't make it impossible for visitors to submit your forms

It is easy to make it impossible for some visitors to submit your website form:

- Use of a background image for the submit button.
- Use of an image for the submit button with no alt attribute (i.e. no label).
- Submitting the form requires the use of Javascript.

As we outlined earlier in this article, you cannot add a label to a background image; so using a background image as the submit button effectively makes that form unusable for people using screen readers or people browsing with images off or browsing with text-only web browsers.

The same invisible submit button effect can also be achieved by using an image but not providing a text alternative for that image, i.e. failing to add the alt attribute to the image code.

Issue with forms and Javascript

One of the most common access issues I've come across is developers making forms inaccessible due to the way they implement Javascript (i.e. to add functionality such as client side checks before submission).

Using Javascript is not a bad idea and does not in itself make a form inaccessible; the issue is how the Javascript is implemented. For example, if running a script is the action required to submit the form then clearly if Javascript is not available the form cannot be submitted.

The simple answer to this issue is to first build your forms in the standard way, so that they work fine without Javascript - and then add any fancy Javascript functionality later; always testing that it still works if Javascript is not turned off.

A few suggested don'ts:

- * Don't use tabindex to force users to tab through form fields in a particular order; design well organised and logical forms instead.
- * Don't use 'Javascript jump menus', i.e. when a user chooses an option from a pull-down menu and are instantly taken to that choice without

having to hit submit, i.e. don't use 'OnChange' in select menus to open a new page or refresh the page. Always provide a submit button as these menus often don't work for keyboard users

In practice: using Contact Form 7 in Wordpress to create more accessible forms

Contact Form 7 is a Wordpress contact form plugin that makes it easy to create forms. However, the default forms produced are not ideal in terms of accessibility.

For example, the plugin automatically adds its' own markup (i.e. paragraph tags and break tags), label tags are not added by default.

First there are a few changes you need to make to Wordpress's configuration file to stop Contact Form 7 adding it's own markup (and it's own style sheet and its' own Javascript).

Add the following code to the file, wp-config.php (it will be in the root folder of your Wordpress install).

```
/* wp-contact-from-7 */  
define ('WPCF7_AUTOP', false);  
define ('WPCF7_LOAD_CSS', false);  
define ('WPCF7_LOAD_JS', false);
```

For more information about Contact form 7 constants see: See <http://contactform7.com/controlling-behavior-by-setting-constants/>

Add labels and add id's to input fields.

When creating a new field you will be presented with the option - among other things - to add an 'id' and a 'placeholder'.

Text field x

Required field?

Name

id (optional) <input style="width: 90%;" type="text" value="address"/>	class (optional) <input style="width: 90%;" type="text"/>
size (optional) <input style="width: 90%;" type="text"/>	maxlength (optional) <input style="width: 90%;" type="text"/>

Akismet (optional)
 This field requires author's name

Default value (optional)
 Use this text as placeholder?

Copy this code and paste it into the form left.

And, put this code into the Mail fields below.

Re: id attribute

Fill in a value for the id field. After you have generated the code for your field manually add your own label for the field and also add a 'for' attribute to the label:

```
<label for="address">Address</label>
[text Address id:address]
```

Notice how the id attribute has been added to the input field, i.e. id:address. You can use this method to manually add id's to other fields.

The value of the 'for' attribute added to the label tag should match the value of the associated id attribute. Here's another example,


```
<label for="yourname">Your Name (required) </label>
```

```
[text* your-name id:yourname]
```

Re: Placeholder text

Don't check the box to add placeholder text. In theory it is a good idea (i.e. it provides contextual information about the form field), however, in practice there are several issues with using this HTML 5 feature:

- * The placeholder text disappears when the field has focus; which can be a problem if it has been used instead of adding a form field label.
- * This presents an issue for people with memory issues and for keyboard users as they must read ahead of the current focus when filling in the form.
- * The default style applied to placeholder text (by Browsers that support it) has very low contrast with the background; (which of course you can fix by adding your own styles).
- * Some users will think the field is already filled in and will skip it.

Add your own HTML to the form

You can edit and add to the text generated by Contact Form 7, i.e. the text in the left-hand form field of the Contact form plugin page. This means you can add your own form grouping code and any additional headings or markup that will make the form easier to use.

Contact Form 7 gets updated often so I recommend you visit the documentation page to check for the latest features: <http://contactform7.com/docs/>

Avoid auto play on videos and audio; audio and videos that start automatically can be very distracting; and for visitors with particular impairments - it can be difficult to figure out how to control the player, i.e. how to stop the annoying audio.

Avoid opening new windows without warning

When adding links - many WYSWYG editors provide the option to open a page in a new window. Don't use it; opening a page in a new window is now very common - but it is still a bad idea.

Granted for many developers they see it as a convention to open links that go to other sites in new windows, however, it can still be confusing for visually impaired users when they discover that their back button doesn't work and they are not sure why (i.e because they didn't notice a new window was opened).

Conclusion

There's lots, lots more to say on each of the topics covered above and about website accessibility in general. However, one of my main aims was to ensure that this document would be useful - but without being too technical or difficult to understand - so I've tried to keep a balance between providing too much details and too little.

Get in touch to tell me if I've succeeded in that aim or not. If not, your feedback will help to improve future versions.

All the best,
Jim Byrne

Accessible Website Design Glasgow

Contact us today. We are hugely experienced award winning web designers and developers. [Read what our clients are saying about how we helped them meet their aims.](#)

Landline: 0141 576 9446

Mobile: 07810 098 119

Email: webdesign@jimbyrne.co.uk