# Creating Accessible Web Forms

**Peter James Rowlett**
**Nottingham Trent University**

peter.rowlett@ntu.ac.uk

**Emma Jane Wright**
**University of Nottingham**

emma@accessingmaterials.org.uk

One of the authors is conducting research into the types of adjustments Higher Education students with a visual or print impairment require in order to access educational materials [1], which includes data collection by quantitative questionnaires. This raises concern that questionnaires may be inherently inaccessible to the group to be investigated. The authors have therefore conducted pilot research into creating accessible web forms.

The techniques investigated have wide application. Any comprehensive sample of the population will necessarily encounter respondents who have a disability or impairment that affects their capacity to use inaccessible web forms. Mann claims that three million people in the UK "*have sight loss, dyslexia or another reading related disability*" [2]. Inaccessible web design also affects users with a fine motor disability, impairing their use of 'standard' computer interface devices, those with a learning difficulty that may impair their comprehension of content and those who for another reason access the Web using a non-'standard' device such as PDA users. Thus an inaccessible web form will exclude a sizeable proportion of the population or lead them to mistakenly input the wrong data, which could damage the integrity of any conclusions.

In addition to this 'good practice' reason for accessible web form design, and the obvious ethical reasons for inclusion [3] there are legal considerations. Since 1 October 1999 there has been a legal obligation for websites to be accessible, and changes to the Disability Discrimination Act 1995 that came into force on 1 October 2004 removed the remaining few exemptions, so those who "*recruit and employ people; provide services; or provide education*" now have an obligation to make "*websites, intranets and extranets accessible*" [4]. This article discusses some of the issues in accessible web form design.

### *Web content accessibility*

Web content accessibility is about inclusiveness. Sir Tim Berners-Lee, the creator of the World Wide Web is famously quoted as saying: "*The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect*" [5]. This goal, laudable, sensible and entirely achievable has nevertheless yet to be realised.

Websites are often based on code that is found to 'work', rather than using code correctly. Bold text is used as headings rather than text marked with 'heading' tags, or the 'blockquote' tag is used to indent text, regardless of whether the text is a quote. The problem is one of semantics; people markup HTML based on how it is displayed (often only in *their* browser), not on how that markup affects the meaning of the content.

Designing accessibly, one should encode the content with the correct semantic meaning and affect the display using Cascading Style Sheets (CSS). Then users with non-'standard' viewing methods can access that meaning.

Here, issues relating particularly to data entry via web forms will be discussed; general accessibility issues are discussed in detail elsewhere (e.g. see [6]).

### *Accessible forms*

**Understanding the form**

If the user misunderstands how data should be entered, they may not be able to complete the form, or may input data incorrectly. If not obviously entered under a miscomprehension, that data might affect the conclusions drawn.

*Question labels:* Certain form elements have a label, e.g. prompting the user to enter data. These labels should be marked using the `label` tag, semantically associating them with the correct form element. Also in modern browsers users click on the label to activate the associated form element, particularly useful with small elements like radio buttons which some might find difficult selecting.

However not all assistive technologies will be configured to use the label association and it is still useful to layout form elements in a standard way. In addition, this ensures consistency, increasing useability. For example, labels should precede text boxes but follow radio buttons [7]. Some elements, such as buttons, will not have a label.

*Language:* Care must be taken to ensure that the clearest possible language is used to improve comprehension.

*Form layout:* It is important that the form be straightforward to navigate and complete. An unusual path through a form can be encoded using the `tabindex` attribute on form elements, but unnecessarily complex organisation of the form should be avoided. A user with an assistive technology will not necessarily be able to access the content in the two-dimensional form a visual display might offer. Care must be taken that structures 'linearise' sensibly; e.g. ensure a table read row-by-row, left to right still has the same meaning. Particularly, linearising a table may dislocate the question prompt and relevant form element. Better still, layout pages using CSS and reserve tables for tabular data, where table header and description tags should be used.

*Logical groupings:* In forms, the `fieldset` tag can be used to group elements with logical association, titled using the `legend` tag. This can enhance understanding of the form structure [7]. For example, a form that asks questions on various topics might have a `fieldset` for each topic; radio buttons might be contained within a `fieldset` to aid the understanding that they are the complete set of options for a question. Browsers may have visual representation of this association, for example a border (see Fig 1) (and this can be altered using CSS).

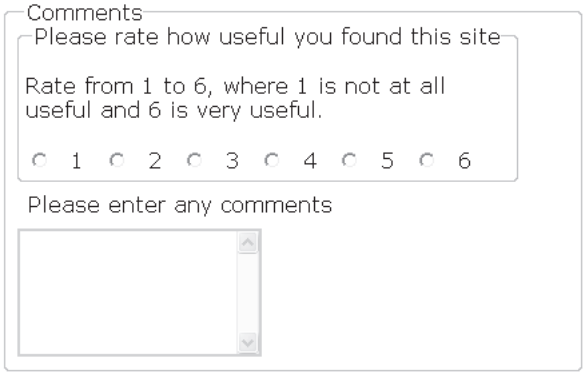


**Fig 1 Visual representation of fieldset groupings**

*Progress:* Awareness of progress through a questionnaire, particularly one on multiple pages, is important. Knowing how many questions remain, users might be less inclined to give up partway through and without knowing how many questions in advance, users may not allow enough time and rush through. This can simply be achieved, by including a note: "*Question/page x of y.*"

Identifying the current position on the page may be useful. CSS can highlight the current element using the `:focus` pseudo-class, though support is not available in all browsers. This can be supplemented using JavaScript to highlight the currently selected form element. Using these methods together will still not, at present ensure support for all users. Still, it may be useful to the users with support.

*Access keys:* Access keys are keys assigned to move the focus to an item (e.g. pressing Alt-s or Apple-s may activate the item assigned access key s). It is important not to present too many access keys to remember. One might assign access keys to the top of the form, the submit button and help page link. It might be useful to assign numeric access keys to questions, so question 1 has access key 1, etc. Some Alt-key combinations have special meaning, for example Alt-f is used to access the Windows File menu, so care must be taken choosing access keys. Access keys might be best assigned to labels, rather than the form elements directly, so a screenreader will read the label out on pressing the access key. Access keys must be listed somewhere, so users are aware of them. Consistency will assist the user in remembering these.

**JavaScript**

Use of JavaScript is not inherently inaccessible. JavaScript can be used to great effect to enhance useability, an important component of accessibility, for example by setting the focus to the first form element when the page loads. Users

without JavaScript are unable to access such enhancements, but care must be taken that they can access basic functionality. A statement that JavaScript is used in this way will tell users without it they are not missing anything vital.

**Automated form validation**

Automated form validation is the process by which data is checked on submission for obvious errors: a question left blank, an answer given outside of a numeric range, etc. Erroneous entries are marked for attention.

This can be achieved using software on the client (the user's machine), such as JavaScript. Errors should be presented one at a time, not all together, as this can be confusing. JavaScript can set the focus to the erroneous entries, greatly increasing useability. However, users without the software will submit data that has not been validated.

Alternatively, server-side software can be used. As needed, the form is reloaded with erroneous entries highlighted (explanatory text may be added). This is completed on the server, and does not require the client to run any particular software. Note that highlighting entries should not rely on colour alone.

Client- and server-side validation can be conducted in parallel; users without the client software will 'fall though' to the server-side validation.

Note that users may not be able to quickly scan for erroneous entries. Indeed, in some access mechanisms it may not be clear that the same page has been reloaded; the user may believe they have progressed to another page. Then, text should be included at the start of the page explaining the errors with intra-page links to the erroneous entries.

**Style**

A properly marked up web form should contain no information about the particular style of the page, this being controlled by CSS. Then, a form can easily offer multiple style sheets (e.g. default, large font, high contrast).

***Specific question types***

*Free text entry*: Text boxes often contain fixed size text, meaning text entered does not resize like the other text on the webpage. A relative font size can be assigned to the text box using CSS, as with paragraph text, ensuring this will enlarge as needed (see Fig 2). A default value in a text box can help assistive technologies identify the text box, though Foskett [8] claims this is no longer required.

*Pulldown list*: Again, the list can be difficult to read and this can be fixed with CSS relative font size. It can be difficult to view all of the options by scrolling through a pulldown list; consider if using radio buttons might be preferable. The `optgroup` tag can be used to group items within a pulldown list, especially important with long and complicated lists.

*Radio buttons and checkboxes*: These can be small, making them difficult to see and select. The size of these elements can be affected in some browsers by assigning them a width and a height using CSS. Using a relative unit, such as `em`, allows the radio buttons to be resized. The input tag, which is used to define radio buttons and checkboxes, is used by other form elements such as text boxes and buttons, which would be affected by the width and height. The radio button and checkbox form elements can be assigned a CSS class, and this class used to resize them only. However, this is in fact a bug and not valid CSS [9]. A label tag should be used so clicking the label selects the relevant radio button or checkbox. If horizontal radio button lists must be used, there must be enough space between options to distinguish them. Vertical radio button lists are clearer.

*Questions arranged in a complex structure*: For example, questions that ask respondents to rank items that are arranged in a table by selecting radio buttons or checkboxes. These question types are often strictly dependant on their visual structure. Enlarging these elements can cause the formatting to become corrupted. In addition, users accessing the page in a linear format can easily lose the meaning of this two-dimensional structure. It may be more sensible to ask each question separately.

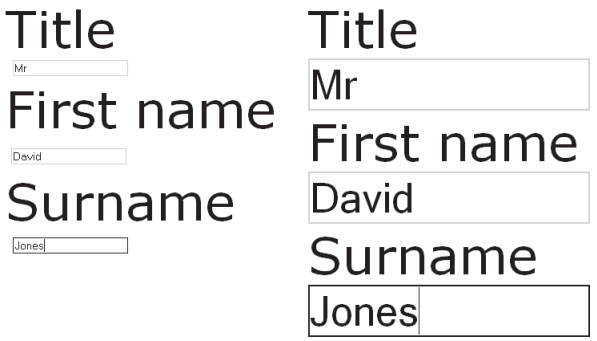


**Fig 2 On the left, the text box input has not enlarged. On the right, the text box input is allowed to resize with the page text.**

***Validating web content accessibility***

Validating for web content accessibility is an important part of determining that content can be accessed. There are automated validation systems that can help in this determination.

Perhaps the best known accessibility checker is Bobby [10], though others are available. Another important step in validating accessibility is to check that the code you are using is valid. The W3C Markup Validation Service can validate webpages and style sheets [11].

In addition, there are issues that cannot be checked automatically. For example, content should be written to be accessible to the widest possible audience. ALT tags must provide a textual alternative to images which conveys the same information as the image. These kinds of issue must be judged by the content producer. Automated validation must then be supplemented with informed manual checking.

Once a theoretically accessible page has been produced, it is vital to ensure it is actually useable by a variety of users. This can be checked by accessing the page with various browsers, operating systems and assistive technologies. It is important to have someone detached from the subject assess the pages. Ideally, a range of users should perform an evaluation.

Since people experience disabilities and impairments in very different ways, it is important to realise that no matter what checks have been made a page still may not be accessible to all users. Absolutes like "*the website is accessible*" should be avoided as practically impossible to justify. Instead, a statement like "*every effort has been made to make sure the website is accessible*" should be preferred. A live page should include the opportunity for users to report accessibility difficulties, and a maintenance system must be in place to quickly accommodate these.

### *Conclusions*

Provided simple rules are obeyed, accessible form design need not be any more difficult than inaccessible design. Accessible design may increase the ease of use and understanding of the form for all users (not only those with disabilities), so it may increase the response rate. Excluding a section of the population from a study by not providing an accessible questionnaire is certainly illegal, and may well call into question the validity of the sample. Can your sample be representative of the entire UK population if several million people are fundamentally denied access?

Discussion here has avoided, where possible, technical detail. Practical tutorials on accessible web forms are available (e.g. see [12]).

### *References*


[1] *Accessing Materials:* http://www.accessingmaterials.org.uk/

[2] Mann, D. *Written off: The latest bulletin on the Right to Read campaign* [online]. Royal National Institute for the Blind, 2004. At: http://www.rnib.org.uk/xpedio/groups/public/documents/publicwebsite/public_writtenoffword.doc [Accessed: 15 July 2005].

[3] *Web Access Centre: The ethical case for web accessibility* [online]. Royal National Institute for the Blind, 2004. At: http://www.rnib.org.uk/xpedio/groups/public/documents/PublicWebsite/public_ethicalcase.hcsp [Accessed: 20 July 2005].

[4] *Web Access Centre: UK Law* [online]. Royal National Institute for the Blind, 2004. At: http://www.rnib.org.uk/xpedio/groups/public/documents/PublicWebsite/public_legalcase.hcsp [Accessed: 19 July 2005].

[5] *Web Accessibility Initiative (WAI)* [online]. World Wide Web Consortium, 2005. At: http://www.w3.org/WAI/ [Accessed: 17 July 2005].

[6] W3C Web Accessibility Initiative. *Getting Started: Making a Web Site Accessible* [online]. World Wide Web Consortium, 2002. At: http://www.w3.org/WAI/gettingstarted/ [Accessed: 17 July 2005].

[7] Lloyd, I. *Accessible HTML/XHTML Forms* [online]. The Web Standards Project, 2004. At: http://www.webstandards.org/learn/tutorials/accessible-forms/01-accessible-forms.html [Accessed: 6 July 2005]

[8] Foskett, M. *Accessible forms: Guidelines, examples and accessible JavaScript tricks* [online]. WebSemantics.co.uk, 2005. At: http://www.websemantics.co.uk/tutorials/accessible_forms/ [Accessed: 21 July 2005].

[9] Korpela, J. *Affecting the presentation of form fields on Web pages* [online]. IT and communication, 2005. At: http://www.cs.tut.fi/~jkorpela/forms/present.html [Accessed: 21 July 2005].

[10] *Bobby: Testing Web Content Accessibility* [online]. Watchfire, 2005. At: http://www.watchfire.com/products/desktop/bobby/default.aspx [Accessed: 17 July 2005].

[11] *W3C Markup Validation Service:* http://validator.w3.org/.

[12] Adams, C. *Accessible, stylish form layout* [online]. The Man in Blue, 2004. At: http://www.themaninblue.com/writing/perspective/2004/03/24/ [Accessed: 20 July 2005].