chapter thirteen

Working with Browsers Part III: Typography

Along with positioning and color, typography is an essential tool of design. Print designers spend years studying the history and application of type. They learn to distinguish between faces that, to the uninitiated, look almost identical, such as Arial and Helvetica. When these traditionally educated designers come to the web, with its limited and contradictory typographic toolsets, they have often been less well equipped to navigate its rocky shoals than those from a non-traditional design background.

Size Matters

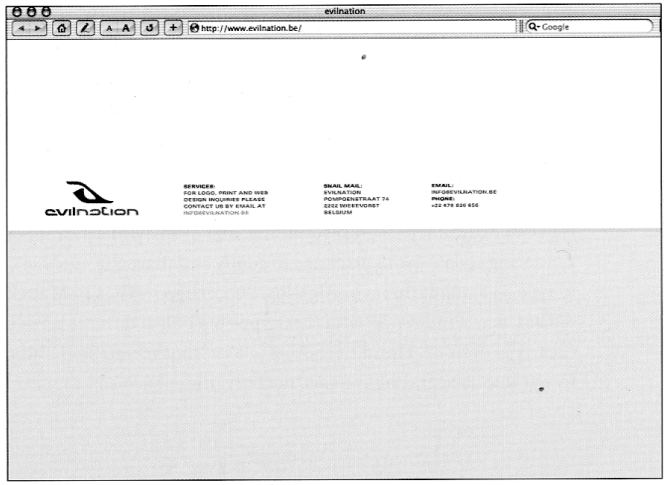
Windows, UNIX, and Macintosh systems come with different installed fonts, at different default resolutions, and often with different default rendering styles—from pixelated to gently antialiased (as in Mac OS 9) to type so smooth that it borders on Photoshop text (as in Mac OS X by default, and Windows XP with Cleartype—but only if the user turns on the Cleartype option). The old `<font size=3>` thus means something different among operating systems, not only in size but also in appearance.

User Control

In addition to variance between platforms, outdated methods, and CSS renderings, the web differs from print in that the user is supposed to retain some control over what she sees. Accommodating the user is as tricky with standards as it is with old-school methods, due to problems discussed in this chapter. It is also difficult for traditionally trained designers to accept the premise of user control. In this chapter, we will discuss the history and problems of web typography and study methods of setting text via web standards.

Old-School Horrors

Tim Berners-Lee, the inventor of the web and the founder of the W3C, viewed his creation as a medium for the easy exchange of text documents; therefore, he included no typographic control in the structured language of HTML. As described in the section "Jumping Through Hoops" in Chapter 2, web designers initially used `<tt>`, `<pre>`, `<blockquote>`, and semantically meaningless paragraph tags to vary typefaces, achieve positioning effects, and simulate leading. They next began using GIF or Flash images of text, a practice that continues to this day [**13.1**, **13.2**], often at the hands of skilled designers who have difficulty accepting the limitations of CSS and XHTML and the trade-offs that are inherent in balancing user versus designer needs.



13.1
Evilnation.be (`www.evilnation.be`) circa 2002. Every "word" on this page was a Flash vector, not real text.

13.2
Still vector after all these years: the same site in 2006 (`www.evilnation.be`). Skilled designers who care about typography often have difficulty accepting the limitations of XHTML and CSS and the trade-offs of user control.

By 1995, with commercial sites springing up right and left and designers hacking HTML to ribbons, something had to be done to provide at least a few basic typographic tools. Netscape gave us the `<font>` tag whose attribute was `size`. You could specify numbers (`<font size=2>`) or relative numbers based on the user's default (`<font size=+1>`). Designers quickly abandoned paragraph tags and other structural elements and controlled their layouts by combining `<font size...>` with `<br>` tags. Not to be outdone, Microsoft gave us `<font face...>`.

Most readers of this book will remember those days and will also recall the problems—chiefly, those of platform difference. Windows assumed a default base size of 16px at 96ppi (pixels per inch). Mac OS, closely tied to print design, assumed a size of 12px at 72ppi based on the PostScript standard. Font sizes that looked dandy on one platform looked too big or too small on the other.

"Stupid Windows," said the Mac-based designers.

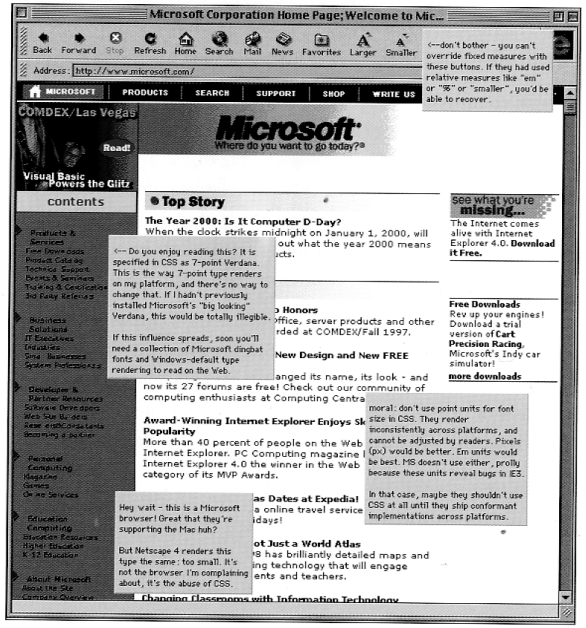
"Stupid Macintosh," said the Windows-based designers.

Points of Difference

Next came early CSS implementations such as that of IE3, thrusting the cross-platform problem into even greater relief. Points (pts) are a unit of print, not of the screen, but designers are familiar with points, and many chose to specify their web text using this unit. In the Windows world, 7pt type was 9px tall, which is the lowest threshold of legibility. On the Mac, 7pt type was 7px tall, making it illegible, and as useless as a beard on a baby.

In 1997, Microsoft.com chose 7pt type [**13.3**] to ballyhoo the CSS prowess of their new IE3 browser for Windows and Macintosh. This was like inviting folks to a movie premiere and de-focusing the projector before screening. The type was equally illegible in IE4 and Netscape 4 on the Macintosh, not because of browser problems, but because of platform differences. Todd Fahrner, soon to be the father of DOCTYPE switching (see Chapter 11), posted Figure **13.3** on his personal site, annotating it to show that points were a useless unit of CSS in terms of screen design (although they are fine for print style sheets).

13.3
In this mid-1990s screen shot, Todd Fahrner documented problems of CSS misuse including screen type specified in points (`style.cleverchimp.com/font_size/points/font_wars.GIF`).



Fahrner also pointed (sorry) out that text set in points and pixels could not (at the time) be resized via the browser's built-in font-size adjustment widget. This was not because CSS1 considered points and pixels fixed units. (CSS1 did not define *any* sizing method as being nonresizable.) It was simply a decision made by the first browser makers to begin supporting CSS. The user *could* reset type set in ems and percentages, but IE3, IE4, and Netscape 4 all suffered from horrendous bugs where type set in ems or percentages was concerned. Fahrner would soon propose a solution to at least some of these problems.

At the time the screen shot shown in Figure **13.3** was posted, the only size unit that worked the same way across browsers and platforms was the nonresizable pixel. Ten years later, the only entirely reliable unit is still the pixel, and it is still nonresizable in IE/Windows (although upcoming IE7 will finally allow text set in pixels to scale).

A Standard Size at Last

In an effort to transcend platform differences, the makers of Netscape and Microsoft's browsers and Mozilla put their heads together in late 1999 and decided to standardize on a default font-size setting of 16px/96ppi across platforms. By putting all platforms on the same page, as it were, browser makers and users could avoid the problems of cross-platform size differences and cruelly useless, illegible text.

The 16px/96ppi Font-Size Standard

On a W3C mailing list in 1998, the plucky Todd Fahrner puckishly proposed standardizing on a 16px default per Windows usage. His recommendation was adopted by all leading browser makers in the year 2000. Although Fahrner's concerns were practical in nature (he wanted to ensure that type could be read online), in the quotation that follows, he refers to something that might strike you as bizarre.

The framers of CSS1 were bound to a pet abstraction wherein the "average" length of a web user's arm was essential to defining the size of a pixel. No, really. Anyway, in advancing his idea about a standardized cross-platform

continues

K2 NAVIGATION BAR

K2.1 (www.epicurious.com, June 16, 2006)

Figure K2.1

Epicurious has a top-running navigation bar—one that runs across the top of the page—on its Web site.

BACKGROUND

From MULTIPLE WAYS TO NAVIGATE (B1), we know we need a consistent interface for browsing and searching a site. The BROWSABLE CONTENT (B2) and SEARCH ACTION MODULE (J1) patterns provide the fundamentals for building a browsable structure and straightforward search. Navigation bars are a common way to provide access to the main parts of your Web site. This pattern describes how to create navigation bars that your customers will find useful.

PROBLEM

Customers need to be able to reach the most important parts of your Web site in a structured, organized way that is easy to understand and use.

Large-scale Web sites need a clear and systematic scheme to make it easy for visitors to navigate them. Web sites on this scale are usually organized into subsites that focus on a specific topic, or into categories that focus on a specific product or type of information.

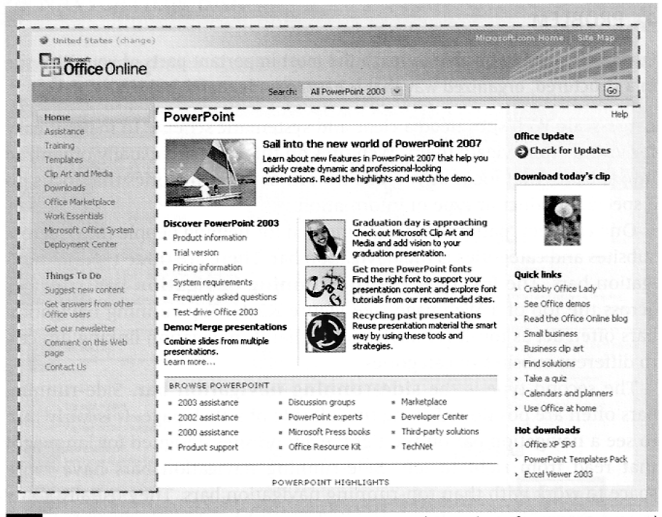
One common pattern that has emerged for helping people move across subsites and categories is the navigation bar. There are three types of navigation bars. The first type, the **top-running navigation bar,** stretches across the top of a Web page (see Figure K2.1). Top-running navigation bars often act as top-level navigation—that is, navigation linking directly to different subsites or categories.[1]

The second type is the **side-running navigation bar.** Side-running bars often are positioned along the left side of a Web page. It is fairly rare to see a navigation bar on the right, even on sites designed for languages that read from right to left. Side-running navigation bars have more space to work with than top-running navigation bars. They usually show more categories, too, often providing second-level navigation that provides links within a subsite.

The third type of navigation bar is the **top-and-left navigation bar,** which resembles an upside-down letter *L* and is sometimes referred to as an *inverted L*. This bar runs across the top and along the left side of a Web page. Often the top-running portion provides broad navigation across subsites, and the side-running portion provides deep navigation within the current subsite (see Figure K2.2).

Navigation bars link to the most important portions of a Web site either through text links or through icons and text. Icons by themselves are usually not effective because they're not always universally understood across cultures or even within a culture. It helps to have a text description to augment an icon (see Figure K2.3).

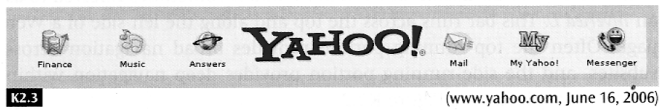
1 Here the word *top* is being used in two different ways. *Top-running* refers to the fact that the navigation bar is positioned across the top of a Web page. *Top-level* means that the navigation bar provides access to all of the major portions of a site.



K2.2 (www.microsoft.com, June 16, 2006)

Figure K2.2

Microsoft's Web site has a navigation bar that runs across both the top and the left. The top part of the bar contains a search action module and links to help and to the site map. The left part contains links to specific topics in the site.



K2.3 (www.yahoo.com, June 16, 2006)

Figure K2.3

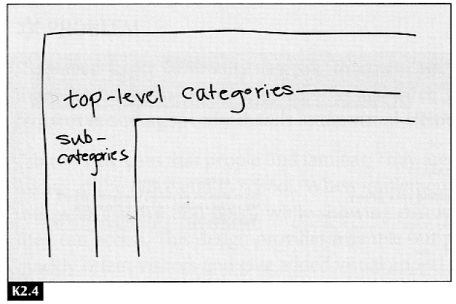
The text under the icons here makes it clear what the images represent.

SOLUTION

Coordinate top-level and second-level navigation in a navigation bar along the top and/or left side of each Web page. Use text, or both icons and text, as links inside the navigation bar.

Figure K2.4

Create a navigation bar that runs along the top and/or left side.



K2.4

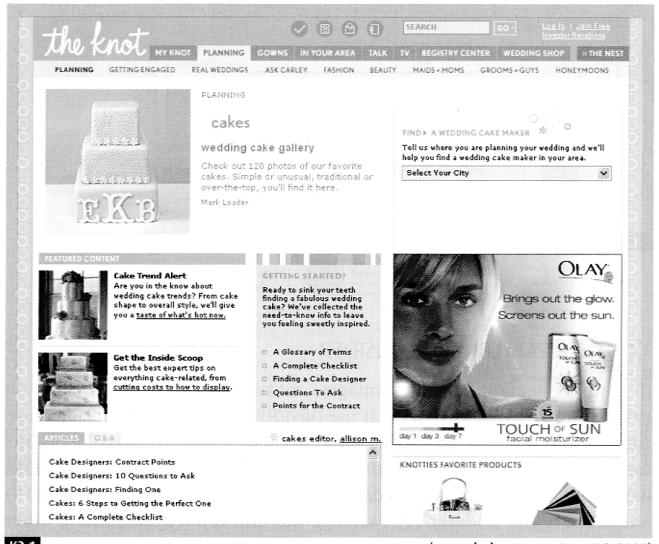
OTHER PATTERNS TO CONSIDER

If your navigation bar uses images, make sure they're FAST-LOADING IMAGES (L2). Navigation bars that use OBVIOUS LINKS (K10) are the clearest. Alternatively, by using HTML POWER (L4) to create a background color distinct from the body copy background color, and by keeping the navigation bar to the left or the top, you can bend the rules of OBVIOUS LINKS (K10) and remove the underlines or change the link color to something other than blue.

TAB ROWS (K3) are often used for top-level navigation. One advantage of tab rows is that they clearly show customers which category they're currently viewing.

CLEAR FIRST READS (I3) are sometimes placed at the top left corner of a Web page, affecting the placement of navigation bars.

K3 TAB ROWS



K3.1 (www.theknot.com, June 16, 2006)

Figure K3.1

Well-designed tab rows provide visual cues that help customers recognize and use them more effectively.

BACKGROUND

MULTIPLE WAYS TO NAVIGATE (B1), BROWSABLE CONTENT (B2), and NAVIGATION BARS (K2) show the value of providing clear and consistent navigation indicators. One type of NAVIGATION BAR (K2) is a tab row, which offers clear visual cues about what your customer can click on, as well as straightforward feedback about the currently selected item.

B1 B2 K2 K2

PROBLEM

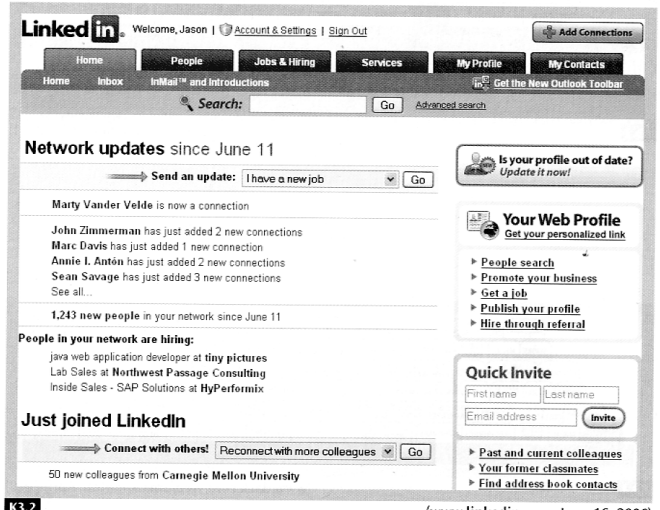
Sites need to let customers navigate through categories of content and give them feedback on where they are. To make tab rows work well, however, requires including specific details in the visual elements.

Tab rows are cues that people find familiar. They are reminiscent of tabs on file folders at the office and in school. When implemented well, they clearly indicate what's active and open, while showing customers which other sections they can access. This design provides a simple but powerful navigation aid to quickly orient visitors and give added visual appeal to a site. Making tab rows work on the Web, though, involves creating some specific graphical devices.

Clearly Identify the Active Tab • Customers need to be able to see which tab is active. Make the active tab stand out from nonactive tabs by giving it a different color and accentuating the contrast in brightness for visitors who cannot distinguish by color differences alone. Making a tab active, even when customers first come to a site, will be another visual clue that the row of rectangles is a collection of tabs. Use color, contrast, and preselection to make active tabs stand out (see Figure K3.2).

Figure K3.2

On LinkedIn's Web site, the active tab is differentiated by color and contrast, and the **Home** tab is preselected when a customer first arrives at the site.

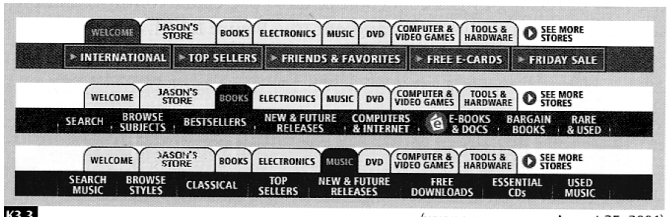


K3.2 (www.linkedin.com, June 16, 2006)

Create an Indicator Line • To reinforce the illusion that the row of tabs is just that, and to bring content to the foreground, give visitors an indicator showing that the content of the page is controlled by tabs. Showing a line below the active tab in the same color—one that extends from left to right over all the content—indicates that the tab is not only a switch, but also a control over the content that customers see (see Figure K3.3).

Tab Rows Have Limitations • Tab rows can do only so much. The number of categories that a tab row can effectively manage on one line is about ten, and when you use multiple rows of tabs the screen begins to look cluttered—too much of it consumed by tabs (see Figure K3.4). An alternative design is to combine a tab row with a JUMP MENU (K16), as shown in Figure K3.5.

K16

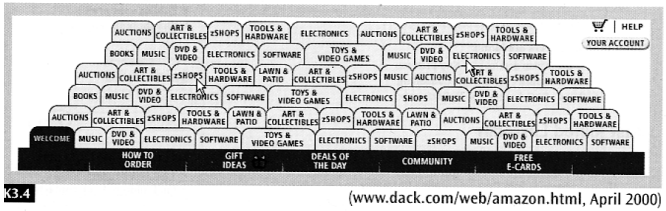


K3.3

(www.amazon.com, August 25, 2001)

Figure K3.3

Although Amazon.com no longer uses multiple individual tabs for its product categories, this figure is still a good example of how to make an effective tab row. The indicator line covers the width of the page, giving the impression that all content on the page belongs to that tab.



K3.4

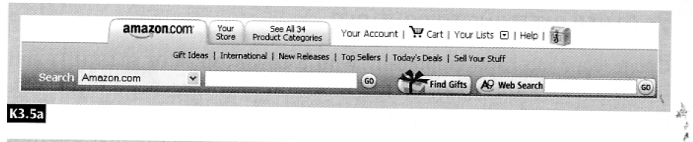
(www.dack.com/web/amazon.html, April 2000)

Figure K3.4

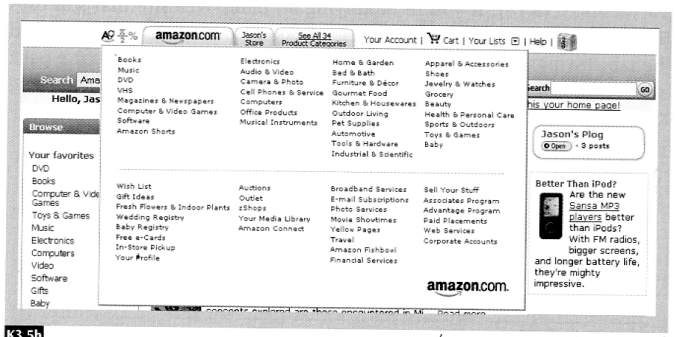
This is an extreme (and fictional) example, but it makes a point: Too many tabs clutter the screen.

Figure K3.5

(a) The most recent version of Amazon.com's tab rows has dramatically cut down on the number of tabs, instead (b) placing them in a floating window that appears when you mouse over or click on a tab.



K3.5a



K3.5b

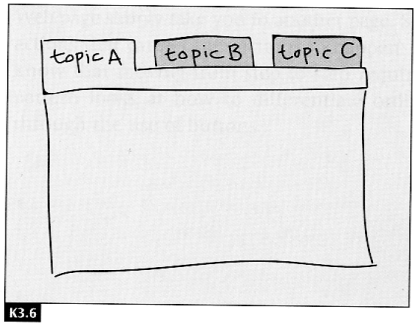
(www.amazon.com, June 16, 2006)

SOLUTION

Create tab rows using an active tab and indicator line, but with no more than ten items, or whatever can fit on one line of tabs. Differentiate an active tab by color and contrast, as well as through preselection. Include an indicator line that extends across the page to create the impression that the whole page below the line belongs to the active tab.

Figure K3.6

Use tab rows to let customers navigate through different categories of information.



K3.6