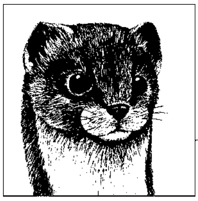

Font and Text Properties

Cascading Style Sheets offer a degree of control over text formatting that approaches desktop publishing. This certainly comes as a relief after years of misusing HTML markup for presentation purposes. Controls for specifying fonts and text formatting are undeniably the most popular use of style sheets and they are the properties that browsers support the most reliably.

This chapter discusses the challenges of typography on the Web and introduces the following text-related CSS 2 properties:

```
font-family          text-decoration      letter-spacing
font-size            text-transform       word-spacing
font-weight          line-height          white-space
font-style           text-indent          direction
font-variant         text-align           unicode-bidi
font                 vertical-align
```

Text color is discussed in the "Foreground Color" section of Chapter 20.

Typography on the Web

For those accustomed to print, the Web offers some unique challenges, usually requiring the relinquishing of control. Typography is a prime example. In print, designers may choose a typeface and point size for headlines and body copy, and as long as the proper font is provided when the printed piece is output, everything looks just the way the designer intended. On the Web, it's not so easy.

Font Issues

Specifying fonts for use on web pages is made difficult by the fact that browsers are limited to displaying fonts that are already installed on the user's local hard drive. So, even though you've specified text to be displayed in the Frutiger font, if users do not have Frutiger installed on their machines, they will see the text in whatever their default browser font happens to be. Fortunately, CSS allows you to specify a list of alternative fonts if your first choice is not found (as discussed in the section "Font Family").

This problem is compounded by the fact that fonts are named inconsistently across platforms and based on the foundry they come from. So even though you want text to show up in plain Times, the font name for that typeface may be Times New Roman or TimesNR or Times Roman. Browsers don't know the difference. This makes it difficult to find a font face even if it (or something like it) is in fact there.

Type Size Issues

The other web typography challenge is type size. Size is problematic due to varying screen resolutions and different default font sizes built into browsers and operating systems. What looks perfectly fine on your monitor may be too small to read for someone else. On top of that, to keep content accessible, text should be sized in a way that allows the end user to resize it (usually larger) to meet special needs. The specific problems of setting text size along with recommendations will be covered in the upcoming "Font Size" section, but for now, suffice it to say that it is not as straightforward as print. It requires knowledge of the medium and occasionally some tough decisions.

Alternatives to Browser Text

Although CSS offers far more control over text formatting than any presentational HTML hack, keep in mind that it is still working in an environment that is somewhat hostile to—or, at the very best, naïve about—typography. From the Web's earliest days, there have been efforts to circumvent the limitations and achieve beautiful typography on web pages. After more than 10 years of trying, there is still no ideal solution, but there are a few options to be aware of.

Text in graphics

It didn't take long for designers (this author included) to start replacing ugly browser text with text set in an inline graphic. For a while, it was not uncommon to run across sites with all every last word of their "content" sunk into a graphic. While this may achieve the short-term goal of preserving the intended font design, it comes at a steep cost. Not only does it increase the file size of the page, but the content is essentially removed from the document. Alternative text (using the `alt` attribute) helps, but does not solve the problem.

Image-replacement techniques

In modern, CSS-based web design, there is a new way to replace text with an image that preserves the text in the source document. There are several variations, but all image-replacement techniques are based on applying the image as a background in the text element and then finding a way to hide the text using CSS. The various image-replacement techniques are covered in detail in Chapter 24.

sIFR text

One of the most interesting web typography solutions to come along is *sIFR*, which stands for Scalable Inman Flash Replacement. It draws inspiration from the image-replacement techniques that were growing popular in CSS-based designs, but uses small Flash movies instead of bitmapped GIF, JPEG, or PNG images. The advantage is that text in Flash movies is vector-based, so it is smooth, anti-aliased, and able to resize with the page. Using a combination of CSS, JavaScript, and Flash technology, sIFR allows authors to "insert rich typography into web pages without sacrificing accessibility, search engine friendliness, or markup semantics."

sIFR (in Version 2.0 as of this writing) was created by Mike Davidson, who built upon the original concept developed by Shaun Inman (the "I" of sIFR). Here's how the process works (taken from the official sIFR site at *www.mikeindustries.com/sifr*).

1. A normal (X)HTML page is loaded into the browser.
2. A JavaScript function is run that first checks that Flash is installed and then looks for whatever tags, IDs, or classes you designate.
3. If Flash isn't installed (or obviously if JavaScript is turned off), the (X)HTML page displays as normal and nothing further occurs. If Flash is installed, JavaScript traverses through the source of your page, measuring each element you've designated as something you'd like "sIFRed."
4. Once measured, the script creates Flash movies of the same dimensions and overlays them on top of the original elements, pumping the original browser text in as a Flash variable.
5. ActionScript inside of each Flash file then draws that text in your chosen typeface at a 6 point size and scales it up until it fits snugly inside the Flash movie.

In optimal browser conditions, this all happens in a split-second, so all of the checking, replacing, and scaling is not visible to the user. Some browsers may struggle with sIFR.

sIFR is not perfect, but it is a promising technique that could lead to more powerful typography solutions. To find out more about sIFR, visit *www.mikeindustries.com/sifr*. There is also an interesting historical document with the history of web typography and the first release of sIFR at *www.mikeindustries.com/blog/archive/2004/08/sifr*.