CHAPTER 13

Styles, Themes, and Master Pages

Using the techniques you've learned so far, you can create polished web pages and let users surf from one page to another. However, to integrate your web pages into a unified, consistent website, you need a few more tools. In this chapter, you'll consider three of the most important tools that you can use: styles, themes, and master pages.

Styles are part of the Cascading Style Sheet (CSS) standard. They aren't directly tied to ASP.NET, but they're still a great help in applying consistent formatting across your entire website. With styles, you can define a set of formatting options once, and reuse it to format different elements on multiple pages. You can even create styles that apply their magic automatically—for example, styles that change the font of all the text in your website without requiring you to modify any of the web page code. Best of all, once you've standardized on a specific set of styles and applied them to multiple pages, you can give your entire website a face-lift just by editing your style sheet.

Styles are genuinely useful, but there are some things they just can't do. Because styles are based on the HTML standard, they have no understanding of ASP.NET concepts like control properties. To fill the gap, ASP.NET includes a *themes* feature, which plays a similar role to styles but works exclusively with server controls. Much as you use styles to automatically set the formatting characteristics of HTML elements, you use themes to automatically set the properties of ASP.NET controls.

Another feature for standardizing websites is *master pages*. Essentially, a master page is a blueprint for part of your website. Using a master page, you can define web page layout, complete with all the usual details such as headers, menu bars, and ad banners. Once you've perfected a master page, you can use it to create *content pages*. Each content page automatically acquires the layout and the content of the linked master page.

By using styles, themes, and master pages, you can ensure that all the pages on your website share a standardized look and layout. In many cases, these details are the difference between an average website and one that looks truly professional.

Styles

In the early days of the Internet, website designers used the formatting features of HTML to decorate these pages. These formatting features were limited, inconsistent, and sometimes poorly supported. Worst of all, HTML formatting led to horribly messy markup, with formatting details littered everywhere.

The solution is the CSS standard, which is supported in all modern browsers. Essentially, CSS gives you a wide range of consistent formatting properties that you can apply to any HTML element. Styles allow you to add borders, set font details, change colors, add margin space and padding, and so on. Many of the examples you've seen so far have in this book have used CSS formatting.

In the following sections, you'll learn the basics of the CSS standard. You'll see how web controls use CSS to apply *their* formatting, and you'll learn how you can explicitly use styles in your ASP.NET web pages.

Style Types

Web pages can use styles in three different ways:

- *Inline style*: An inline style is a style that's placed directly inside an HTML tag. This can get messy, but it's a reasonable approach for one-time formatting. You can remove the style and put it in a style sheet later.
- *Internal style sheet*: An internal style sheet is a collection of styles that are placed in the <head> section of your web page markup. You can then use the styles from this style sheet to format the web controls on that page. By using an internal style sheet, you get a clear separation between formatting (your styles) and your content (the rest of your HTML markup). You can also reuse the same style for multiple elements.
- *External style sheet*: An external style sheet is similar to an internal style sheet, except it's placed in a completely separate file. This is the most powerful approach, because it gives you a way to apply the same style rules to many pages.

You can use all types of styles with ASP.NET web pages. You'll see how in the following sections.

Creating a Basic Inline Style

To apply a style to an ordinary HTML element, you set the style attribute. Here's an example that gives a blue background to a paragraph:

```
<p style="background: Blue">This text has a blue background.</p>
```

Every style consists of a list of one or more formatting properties. In the preceding example, the style has a single formatting property, named *background*, which is set to the value Blue. To add multiple style properties, you simply separate them with semicolons, as shown here:

```
<p style="color:White; background:Blue; font-size:x-large; padding:10px">
This text has a blue background.</p>
```

This style creates large white text with a blue background and 10 pixels of spacing between the edge of the element (the blue box) and the text content inside.