We know what your next question is: why should I look at Taconite? With Taconite, you don't have to deal with JavaScript and, more important, making sure it works across multiple browsers. The heart of Taconite is a parser that converts normal HTML code into a series of JavaScript commands that dynamically create the content on the browser.

The Theory of Taconite

Generating dynamic HTML has always been difficult. As we’ve discussed, in the early days of the Web, you had CGI and then servlets. Of course, both of these solutions worked, but at their core, they created HTML content via string concatenation. As any servlet developer will tell you, string concatenation is difficult to develop, tedious, error prone, and hard to maintain.

Template models such as JSP and ASP were created to ease the burden by allowing you to use normal HTML with some special markup to develop dynamic pages. These pages are much easier to build and maintain than their “string concatenation” cousins.

As you've seen in some of our examples, using Ajax to generate dynamic content creates many of the same issues—you can use DOM methods to create content, but this approach can be irksome and often results in lots of extra code. Since you're reading this book, you're better equipped than the average developer; however, trust us when we say some DOM methods behave differently across browsers.

Many people have come to rely on `innerHTML` combined with string concatenation when they decide to put a new Ajax feature in their application. Although building the `innerHTML` content on the server (taking advantage of your favorite language) certainly helps, it still doesn't negate the issues with `innerHTML`. Though widely supported, `innerHTML` is not a W3C standard. You can do a lot with `innerHTML`, but it has its limits. For instance, you can't insert a table row between other table rows. Of course, the biggest challenge, and we're sure this will come as a shock to many of you, is that `innerHTML` behavior varies across browsers.

The Solution

Taconite solves this core issue by providing a custom parser that converts HTML into the equivalent W3C DOM methods used to create the specified content. This means you can write dynamic content using whatever HTML-generating technology with which you're most comfortable. You get the best of both worlds: you get your snazzy new Ajax feature without dealing with string concatenation or using standard W3C DOM calls. And since we're such nice guys, we've taken care of all the nasty cross-browser issues!

This parser is used by a set of custom JSP tags that allow you to write content in a way that is natural—as HTML embedded within a JSP. You no longer have to crowd your pages with a slew of `document.createElement` and `document.appendChild` commands to dynamically create new content. The JSP custom tags working with the parser take care of all this while you sit back and enjoy your refreshing iced mocha! The resulting JavaScript is returned to the browser as part of the response where the client-side JavaScript library takes over and executes these commands to produce the desired output.

Taconite doesn't try to reinvent the wheel; instead, it focuses on the Achilles heel of Ajax development: the potentially tedious task of writing large amounts of JavaScript to dynamically update your Web page. Unlike some toolkits, Taconite doesn't try to invent a new way of sending request data to the server; instead, it relies on the tried-and-true name/value pairs either embedded within the query string of a GET request or tucked safely in the body of a POST. The client-side library even automates the creation of the name/value query string—heck, we