CHAPTER 7

JSDigester: Taking the Pain Out of Client-Side XML

I'll just come out and say it: parsing XML in a browser is not a particularly pleasant experience. Actually, if you think about it, parsing XML *anywhere* can be a bit of a hassle. However, one library that does make it a bearable experience is the Jakarta Commons Digester component (`http://jakarta.apache.org/commons/digester`). Digester allows you to specify a series of rules that will be triggered by various elements in an XML document. These rules may handle the parsing in a number of ways, including creating and populating objects from the XML. Wouldn't it be great if we could do the same thing in JavaScript? Well, we're going to make that dream a reality in this chapter, and in the process, make working with XML on the client a much less painful experience.

Parsing XML in JavaScript

Parsing XML on the client is about as much fun as a gum scraping is for most people. Believe me, I don't make the dental analogy lightly (well, while I've never had my gums scraped, I *am* married with children, so I figure I know what it would feel like). If you've ever done much in the way of parsing XML in JavaScript, then the code shown in Listing 7-1 will probably look both familiar *and* painful.

Listing 7-1. *Parsing XML in JavaScript in a Browser*

```
<html>

  <head>

    <link rel="StyleSheet" href="styles.css" type="text/css">

    <title>Simple JavaScript XML Parsing Example</title>

    <script>

      function doParsing() {
```

```
        // This is the XML we will parse.
        var xml = "<messages>";
        xml += "<msg poster=\"Frank\">Hello!</msg>";
        xml += "<msg poster=\"Traci\">I hope all is well with you!</msg>";
        xml += "<msg poster=\"Andrew\">Well, I guess that's it.</msg>";
        xml += "<msg poster=\"Ashley\">Have a good day!</msg>";
        xml += "</messages>";

        // Instantiate an XML parser (or DOM, depending on browser).
        var xmlDoc = null;
        if (window.XMLHttpRequest){
          var parser = new DOMParser();
          xmlDoc = parser.parseFromString(xml, "application/xml");
        } else {
          xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
          xmlDoc.async = false;
          xmlDoc.loadXML(xml);
        }

        // Now iterate over the DOM created above, and construct an output
        // string to display.
        var strOut = "Root node = " + xmlDoc.documentElement.nodeName + "<br>";
        for (var i = 0; i < xmlDoc.documentElement.childNodes.length; i++) {
          strOut += "nodeName = " +
            xmlDoc.documentElement.childNodes[i].nodeName + ", poster = " +
            xmlDoc.documentElement.childNodes[i].getAttribute("poster") +
            ", text = " +
            xmlDoc.documentElement.childNodes[i].firstChild.nodeValue + "<br>";
        }
        document.getElementById("divOut").innerHTML = strOut;

      }

    </script>

  </head>

  <body class="cssBody">

    <div class="cssTitle">JavaScript XML Parsing Example</div>
    <br><br>
    <input type="button" value="Click me to parse XML"
      onClick="doParsing();" class="cssBody">
    <br><br>
    Info will appear here:
```