11.9 Using Scriptlets to Make Parts of the JSP Page Conditional

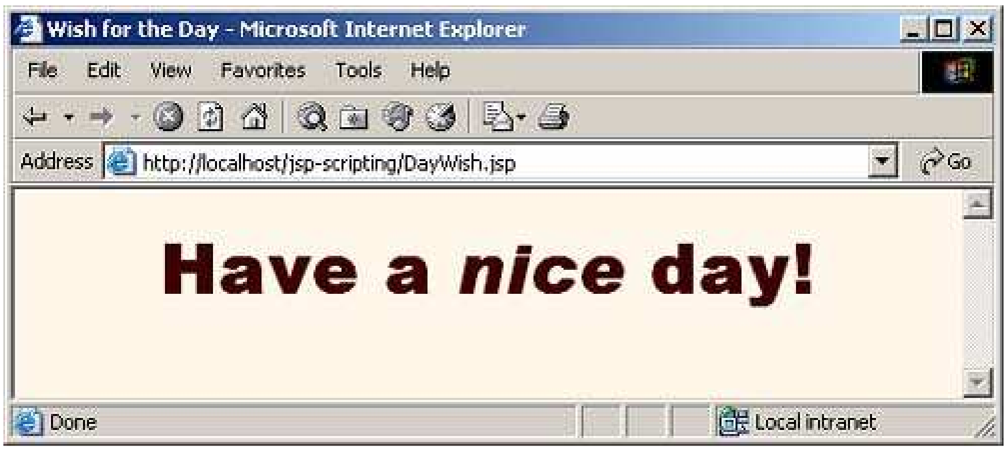
Another use of scriptlets is to conditionally output HTML or other content that is *not* within any JSP tag. Key to this approach are the facts that (a) code inside a scriptlet gets inserted into the resultant servlet's `_jspService` method (called by `service`) *exactly* as written and (b) that any static HTML (template text) before or after a scriptlet gets converted to `print` statements. This behavior means that scriptlets need not contain complete Java statements and that code blocks left open can affect the static HTML or JSP outside the scriptlets. For example, consider the JSP fragment of Listing 11.9 that contains mixed template text and scriptlets.

Listing 11.9 DayWish.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Wish for the Day</TITLE>
<LINK REL=STYLESHEET
      HREF="JSP-Styles.css"
      TYPE="text/css">
</HEAD>
<BODY>
<% if (Math.random() < 0.5) { %>
<H1>Have a <I>nice</I> day!</H1>
<% } else { %>
<H1>Have a <I>lousy</I> day!</H1>
<% } %>
</BODY></HTML>
```

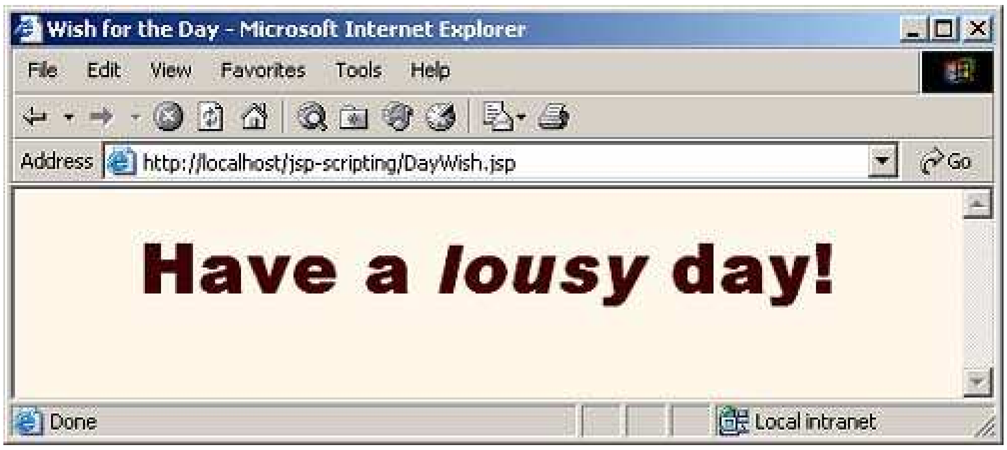
You probably find the bold part a bit confusing. We certainly did the first few times we saw constructs of this nature. Neither the "have a nice day" nor the "have a lousy day" lines are contained within a JSP tag, so it seems odd that only one of the two becomes part of the output for any given request. See Figures 11-8 and 11-9.

Figure 11-8. One possible result of `DayWish.jsp`.


Wish for the Day - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost/jsp-scripting/DayWish.jsp
Go
Done
Local intranet

Have a *nice* day!

Figure 11-9. Another possible result of `DayWish.jsp`.


Wish for the Day - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost/jsp-scripting/DayWish.jsp
Go
Done
Local intranet

Have a *lousy* day!

Don't panic! Simply follow the rules for how JSP code gets converted to servlet code. Once you think about how this example will be converted to servlet code by the JSP engine, you get the following easily understandable result.

```
if (Math.random() < 0.5) {
  out.println("<H1>Have a <I>nice</I> day!</H1>");
} else {
  out.println("<H1>Have a <I>lousy</I> day!</H1>");
}
```

The key is that the first two scriptlets do not contain complete statements, but rather partial statements that have dangling braces. This serves to capture the subsequent HTML within the `if` or `else` clauses.

Overuse of this approach can lead to JSP code that is hard to understand and maintain. Avoid using it to conditionalize large sections of HTML, and try to keep your JSP pages as focused on presentation (HTML output) tasks as possible. Nevertheless, there are some situations in which the alternative approaches are also unappealing. The primary example is generation of lists or tables containing an indeterminate number of entries. This happens quite frequently when you are presenting data that is the result of a database query. See Chapter 17 (Accessing Databases with JDBC) for details on database access from Java code. Besides, even if *you* do not use this approach, you are bound to see examples of it in your projects, and you need to understand how and why it works as it does.