12.3 Conditionally Generating Excel Spreadsheets

In most cases in which you generate non-HTML content with JSP, you know the content type in advance. In those cases, the `contentType` attribute of the `page` directive is appropriate: it requires no explicit Java syntax and can appear anywhere in the page.

Occasionally, however, you may want to build the same content, but change the listed content type depending on the situation. For example, many word processing and desktop publishing systems can import HTML pages. So, you could arrange to have the page come up either in the publishing system or in the browser, depending on the content type you send. Similarly, Microsoft Excel can import tables that are represented in HTML with the `TABLE` tag. This capability suggests a simple method of returning either HTML or Excel content, depending on which the user prefers: just use an HTML table and set the content type to `application/vnd.ms-excel` only if the user requests the results in Excel.

Unfortunately, this approach brings to light a small deficiency in the `page` directive: attribute values cannot be computed at runtime, nor can `page` directives be conditionally inserted as can template text. So, the following attempt results in Excel content regardless of the result of the `checkUserRequest` method.

```
<% boolean usingExcel = checkUserRequest(request); %>
<% if (usingExcel) { %>
<%@ page contentType="application/vnd.ms-excel" %>
<% } %>
```

Fortunately, there is a simple solution to the problem of conditionally setting the content type: just use scriptlets and the normal servlet approach of `response.setContentType`, as in the following snippet:

```
<%
String format = request.getParameter("format");
if ((format != null) && (format.equals("excel"))) {
  response.setContentType("application/vnd.ms-excel");
}
%>
```

For example, we once worked on a project that displayed financial (budget) information to authorized users. The data could be displayed in a table in a regular Web page if the user merely wanted to review it, or it could be placed into an Excel spreadsheet if the user wanted to put it into a report. When we first joined the project, there were two entirely separate pieces of code for each task. We changed it to build the same HTML table either way and to merely change the content type. Voila!

Listing 12.3 shows a page that uses this approach; Figures 12-5 and 12-6 show the results. In a real application, of course, the data would almost certainly come from a database. We use static values here for simplicity, but see Chapter 17 (Accessing Databases with JDBC) for information on talking to relational databases from servlets and JSP pages.

Listing 12.3 ApplesAndOranges.jsp (continued)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Comparing Apples and Oranges</TITLE>
<LINK REL=STYLESHEET
      HREF="JSP-Styles.css"
```

```
      TYPE="text/css">
</HEAD>

<BODY>
<CENTER>
<H2>Comparing Apples and Oranges</H2>
<%
String format = request.getParameter("format");
if ((format != null) && (format.equals("excel"))) {
  response.setContentType("application/vnd.ms-excel");
}
%>
<TABLE BORDER=1>
  <TR><TH></TH>         <TH>Apples<TH>Oranges
  <TR><TH>First Quarter <TD>2307  <TD>4706
  <TR><TH>Second Quarter<TD>2982  <TD>5104
  <TR><TH>Third Quarter <TD>3011  <TD>5220
  <TR><TH>Fourth Quarter<TD>3055  <TD>5287
</TABLE>
</CENTER></BODY></HTML>
```

Figure 12-5. The default result of `ApplesAndOranges.jsp` is HTML content.

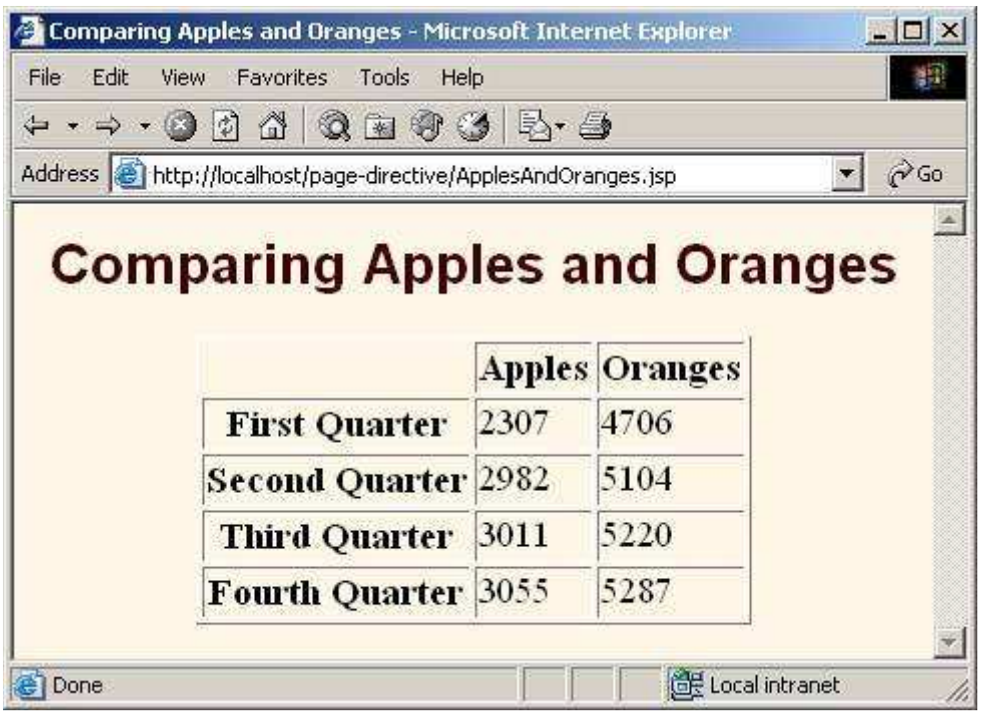
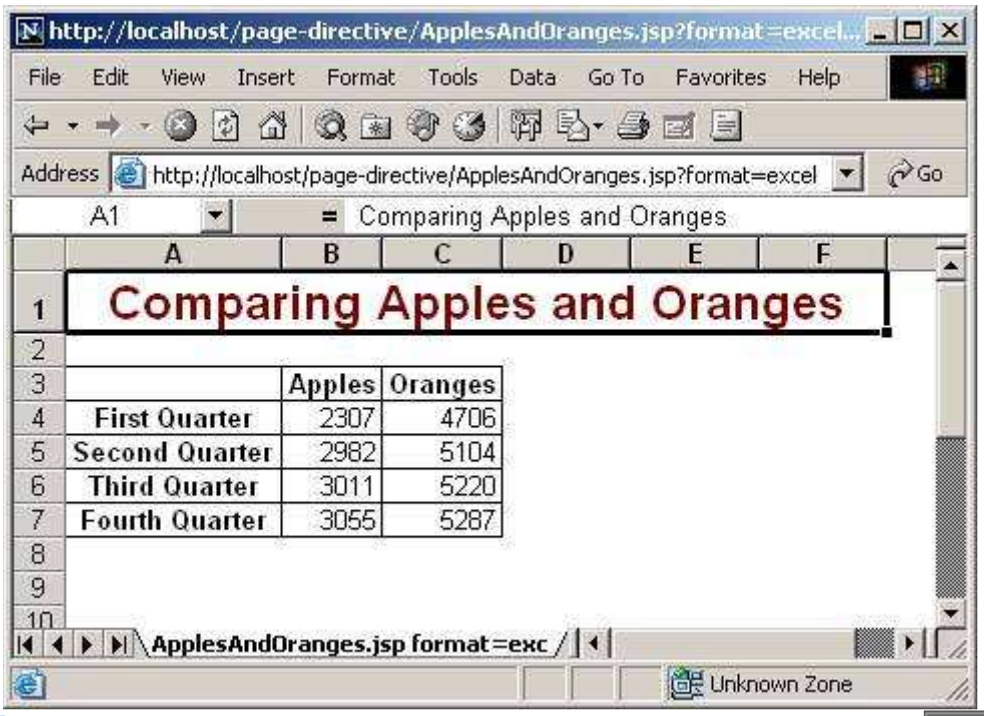


Figure 12-6. Specifying `format=excel` for `ApplesAndOranges.jsp` results in Excel content.



[Team LiB]

PREVIOUS NEXT