11.11 Declaration Example

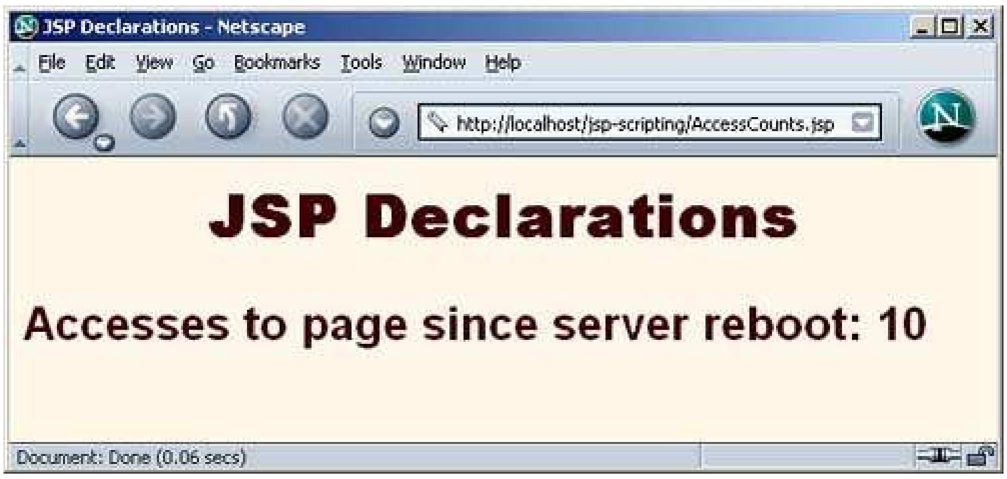
In this example, the following JSP snippet prints the number of times the current page has been requested since the server was booted (or the servlet class was changed and reloaded). A hit counter in two lines of code!

```
<%! private int accessCount = 0; %>
Accesses to page since server reboot:
<%= ++accessCount %>
```

Recall that multiple client requests to the same servlet result only in multiple threads calling the `service` method of a single servlet instance. They do *not* result in the creation of multiple servlet instances except possibly when the servlet implements the now-deprecated `SingleThreadModel` interface (see Section 3.7). Thus, instance variables (fields) of a normal servlet are shared by multiple requests, and `accessCount` does not have to be declared `static`. Now, advanced readers might wonder if the snippet just shown is thread safe; does the code guarantee that each visitor gets a unique count? The answer is no; in unusual situations multiple users could, in principle, see the same value. For access counts, as long as the count is correct in the long run, it does not matter if two different users occasionally see the same count. But, for values such as session identifiers, it is critical to have unique values. For an example that is similar to the previous snippet but that uses `synchronized` blocks to guarantee thread safety, see the discussion of the `isThreadSafe` attribute of the `page` directive in Chapter 12.

Listing 11.12 shows the full JSP page; Figure 11-10 shows a representative result. Now, before you rush out and use this approach to track access to all your pages, a couple of cautions are in order.

Figure 11-10. Visiting `AccessCounts.jsp` after it has been requested nine previous times by the same or different clients.


JSP Declarations - Netscape
File Edit View Go Bookmarks Tools Window Help
http://localhost/jsp-scripting/AccessCounts.jsp
Document: Done (0.06 secs)

JSP Declarations

Accesses to page since server reboot: 10

First of all, you couldn't use this for a real hit counter, since the count starts over whenever you restart the server. So, a real hit counter would need to use `jspInit` and `jspDestroy` to read the previous count at startup and store the old count when the server is shut down.

Second, even if you use `jspDestroy`, it would be possible for the server to crash unexpectedly (e.g., when a rolling blackout strikes Silicon Valley). So, you would have to periodically write the hit count to disk.

Finally, some advanced servers support distributed applications whereby a cluster of servers appears to the client as a single server. If your servlets or JSP pages might need to support distribution in this way, plan ahead and avoid the use of fields for persistent data. Use a database instead. (Note that session objects are automatically shared across distributed applications as long as the values are `Serializable`. But session values are specific to each user, whereas we need client-independent data in this case.)

Listing 11.12 AccessCounts.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>JSP Declarations</TITLE>
<LINK REL=STYLESHEET
      HREF="JSP-Styles.css"
      TYPE="text/css">
</HEAD>
<BODY>
<H1>JSP Declarations</H1>
<%! private int accessCount = 0; %>
<H2>Accesses to page since server reboot:
<%= ++accessCount %></H2>
</BODY></HTML>
```