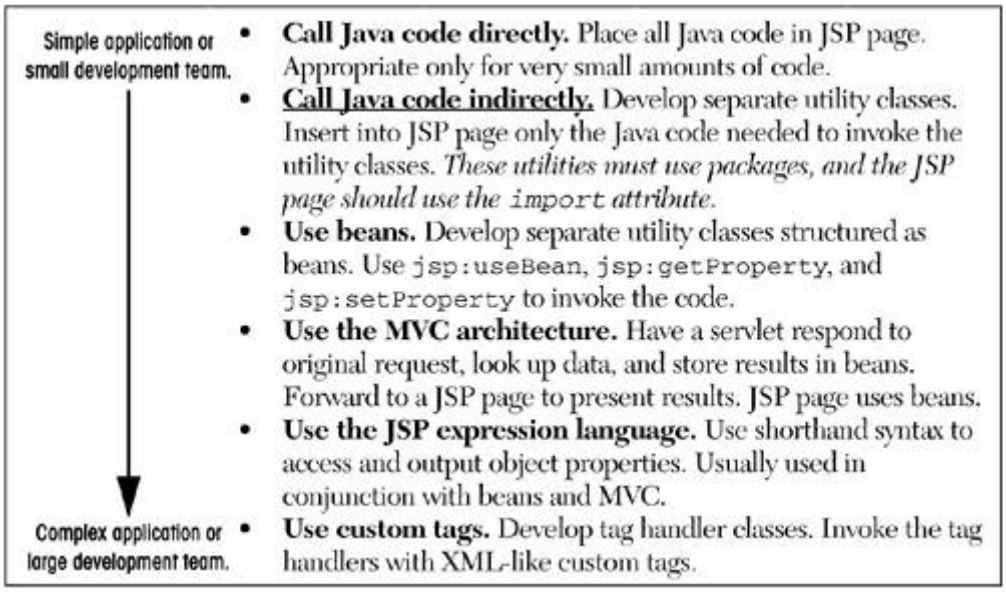
# 12.1 The import Attribute

The `import` attribute of the `page` directive lets you specify the packages that should be imported by the servlet into which the JSP page gets translated. As discussed in Section 11.3 (Limiting the Amount of Java Code in JSP Pages) and illustrated in Figure 12-1, using separate utility (helper) classes makes your dynamic code easier to write, maintain, debug, test, and reuse.

**Figure 12-1. Strategies for invoking dynamic code from JSP.**




When you use utility classes, remember that they should *always* be in packages. For one thing, packages are a good strategy on any large project because they help protect against name conflicts. With JSP, however, packages are absolutely required. The reason is that, in the absence of packages, classes you reference are assumed to be in the same package as the current class. For example, suppose that a JSP page contains the following scriptlet.

```
<% Test t = new Test(); %>
```

Now, if `Test` is in an imported package, there is no ambiguity. But, if `Test` is not in a package or the package to which `Test` belongs is not explicitly imported, then the system will assume that `Test` is in the same package as the autogenerated servlet. The problem is that the autogenerated servlet's package is not known! It is quite common for servers to create servlets whose package is determined by the directory in which the JSP page is placed. Other servers use different approaches. So, you simply cannot rely on packageless classes working properly. The same argument applies to beans (Chapter 14), since beans are just classes that follow some simple naming and structure conventions.

### Core Approach

*Always put your utility classes and beans in packages.*

By default, the servlet imports `java.lang.*`, `javax.servlet.*`, `javax.servlet.jsp.*`, `javax.servlet.http.*`, and possibly some number of server-specific entries. Never write JSP code that relies on any server-specific classes being imported automatically; doing so makes your code nonportable.

Use of the `import` attribute takes one of the following two forms.

```
<%@ page import="package.class" %>
<%@ page import="package.class1,...,package.classN" %>
```

For example, the following directive signifies that all classes in the `java.util` package should be available to use without explicit package identifiers.

```
<%@ page import="java.util.*" %>
```

The `import` attribute is the only `page` attribute that is allowed to appear multiple times within the same document. Although `page` directives can appear anywhere within the document, it is traditional to place `import` statements either near the top of the document or just before the first place that the referenced package is used.

Note that, although the JSP pages go in the normal HTML directories of the server, the classes you write that are used by JSP pages must be placed in the special Java-code directories (e.g., `.../WEB-INF/classes/directoryMatchingPackageName`). See Sections 2.10 (Deployment Directories for Default Web Application: Summary) and 2.11 (Web Applications: A Preview) for information on these directories.

For example, Listing 12.1 presents a page that illustrates each of the three scripting elements from the previous chapter. The page uses three classes not in the standard JSP import list: `java.util.Date`, `coreservlets.CookieUtilities` (see Listing 8.3), and `coreservlets.LongLivedCookie` (see Listing 8.4). So, to simplify references to these classes, the JSP page uses

```
<%@ page import="java.util.*,coreservlets.*" %>
```

Figures 12-2 and 12-3 show some typical results.

## Listing 12.1 ImportAttribute.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>The import Attribute</TITLE>
<LINK REL=STYLESHEET
      HREF="JSP-Styles.css"
      TYPE="text/css">
</HEAD>
<BODY>
<H2>The import Attribute</H2>
<%-- JSP page Directive --%>
<%@ page import="java.util.*,coreservlets.*" %>
<%-- JSP Declaration --%>
<%!
private String randomID() {
  int num = (int)(Math.random()*10000000.0);
  return("id" + num);
}
private final String NO_VALUE = "<I>No Value</I>";
%>
<%-- JSP Scriptlet --%>
```

```
<%
String oldID =
  CookieUtilities.getCookieValue(request, "userID", NO_VALUE);
if (oldID.equals(NO_VALUE)) {
  String newID = randomID();
  Cookie cookie = new LongLivedCookie("userID", newID);
  response.addCookie(cookie);
}
%>
<%-- JSP Expressions --%>
This page was accessed on <%= new Date() %> with a userID
cookie of <%= oldID %>.
</BODY></HTML>
```

**Figure 12-2. `ImportAttribute.jsp` when first accessed.**


**Figure 12-3. `ImportAttribute.jsp` when accessed in a subsequent request.**


[ Team LiB ]

PREVIOUS NEXT