

## 9.5 Encoding URLs Sent to the Client

By default, servlet containers (engines) use cookies as the underlying mechanism for session tracking. But suppose you reconfigure your server to use URL rewriting instead? How will your code have to change?

The goods news: your core session-tracking code does not need to change at all.

The bad news: lots of *other* code has to change. In particular, if any of your pages contain links back to your own site, you have to explicitly add the session data to the URL. Now, the servlet API provides methods to add this information to whatever URL you specify. The problem is that you have to *call* these methods; it is not technically feasible for the system to examine the output of all of your servlets and JSP pages, figure out which parts contain hyperlinks back to your site, and modify those URLs. You have to tell it which URLs to modify. This requirement means that you cannot have static HTML pages if you use URL rewriting for session tracking, or at least you cannot have static HTML pages that refer to your own site. This is a significant burden in many applications, but worth the price in a few.

### Core Warning



*If you use URL rewriting for session tracking, most or all of your pages will have to be dynamically generated. You cannot have any static HTML pages that contain hyperlinks to dynamic pages at your site.*

There are two possible situations in which you might use URLs that refer to your own site.

The first one is where the URLs are embedded in the Web page that the servlet generates. These URLs should be passed through the `encodeURL` method of `HttpServletResponse`. The method determines if URL rewriting is currently in use and appends the session information only if necessary. The URL is returned unchanged otherwise.

Here's an example:

```
String originalURL = someRelativeOrAbsoluteURL;
String encodedURL = response.encodeURL(originalURL);
out.println("<A HREF=\"" + encodedURL + "\">...</A>");
```

The second situation in which you might use a URL that refers to your own site is in a `sendRedirect` call (i.e., placed into the `Location` response header). In this second situation, different rules determine whether session information needs to be attached, so you cannot use `encodeURL`. Fortunately, `HttpServletResponse` supplies an `encodeRedirectURL` method to handle that case. Here's an example:

```
String originalURL = someURL;
String encodedURL = response.encodeRedirectURL(originalURL);
response.sendRedirect(encodedURL);
```

If you think there is a reasonable likelihood that your Web application will eventually use URL rewriting instead of cookies, it is good practice to plan ahead and encode URLs that reference your own site.