6.4 A Front End to Various Search Engines

Suppose that you want to make a "one-stop searching" site that lets users search any of the most popular search engines without having to remember many different URLs. You want to let users enter a query, select the search engine, and then send them to that search engine's results page for that query. If users omit the search keywords or fail to select a search engine, you have no site to redirect them to, so you want to display an error page informing them of this fact.

Listing 6.2 (`SearchEngines.java`) presents a servlet that accomplishes these tasks by making use of the 302 (`Found`) and 404 (`Not Found`) status codes—the two most common status codes other than 200. The 302 code is set by the shorthand `sendRedirect` method of `HttpServletResponse`, and 404 is specified by `sendError`.

In this application, a servlet builds an HTML form (see Figure 6-3 and the source code in Listing 6.5) that displays a page to let the user specify a search string and select the search engine to use. When the form is submitted, the servlet extracts those two parameters, constructs a URL with the parameters embedded in a way appropriate to the search engine selected (see the `SearchSpec` and `SearchUtilities` classes of Listings 6.3 and 6.4), and redirects the user to that URL (see Figure 6-4). If the user fails to choose a search engine or specify search terms, an error page informs the client of this fact (see Figure 6-5, but recall warnings about Internet Explorer under the 404 status code in the previous section).

Listing 6.2 SearchEngines.java

```
package coreservlets;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.net.*;

/** Servlet that takes a search string and a search
 *  engine name, sending the query to
 *  that search engine. Illustrates manipulating
 *  the response status code. It sends a 302 response
 *  (via sendRedirect) if it gets a known search engine,
 *  and sends a 404 response (via sendError) otherwise.
 */


public class SearchEngines extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
      throws ServletException, IOException {
    String searchString = request.getParameter("searchString");
    if ((searchString == null) ||
        (searchString.length() == 0)) {
      reportProblem(response, "Missing search string");
      return;
    }
    // The URLEncoder changes spaces to "+" signs and other
    // non-alphanumeric characters to "%XY", where XY is the
    // hex value of the ASCII (or ISO Latin-1) character.
    // Browsers always URL-encode form values, and the
    // getParameter method decodes automatically. But since
    // we're just passing this on to another server, we need to
    // re-encode it to avoid characters that are illegal in
```

```
    // URLs. Also note that JDK 1.4 introduced a two-argument
    // version of URLEncoder.encode and deprecated the one-arg
    // version. However, since version 2.3 of the servlet spec
    // mandates only the Java 2 Platform (JDK 1.2 or later),
    // we stick with the one-arg version for portability.
    searchString = URLEncoder.encode(searchString);

    String searchEngineName =
      request.getParameter("searchEngine");
    if ((searchEngineName == null) ||
        (searchEngineName.length() == 0)) {
      reportProblem(response, "Missing search engine name");
      return;
    }
    String searchURL =
      SearchUtilities.makeURL(searchEngineName, searchString);
    if (searchURL != null) {
      response.sendRedirect(searchURL);
    } else {
      reportProblem(response, "Unrecognized search engine");
    }
  }

  private void reportProblem(HttpServletResponse response,
                             String message)
      throws IOException {
    response.sendError(response.SC_NOT_FOUND, message);
  }
}
```

Listing 6.3 SearchSpec.java

```
package coreservlets;
/** Small class that encapsulates how to construct a
 *  search string for a particular search engine.
 */

public class SearchSpec {
  private String name, baseURL;

  public SearchSpec(String name,
                    String baseURL) {
    this.name = name;
    this.baseURL = baseURL;
  }

  /** Builds a URL for the results page by simply concatenating
   *  the base URL (http://...?someVar=") with the URL-encoded
   *  search string (jsp+training).
   */

  public String makeURL(String searchString) {
    return(baseURL + searchString);
  }

  public String getName() {
    return(name);
  }
}
```

Listing 6.4 SearchUtilities.java

```
package coreservlets;
```

```
/** Utility with static method to build a URL for any
 *  of the most popular search engines.
 */

public class SearchUtilities {
  private static SearchSpec[] commonSpecs =
    { new SearchSpec("Google",
                     "http://www.google.com/search?q="),
      new SearchSpec("AllTheWeb",
                     "http://www.alltheweb.com/search?q="),
      new SearchSpec("Yahoo",
                     "http://search.yahoo.com/bin/search?p="),
      new SearchSpec("AltaVista",
                     "http://www.altavista.com/web/results?q="),
      new SearchSpec("Lycos",
                     "search.lycos.com/default.asp?query="),
      new SearchSpec("HotBot",
                     "http://hotbot.com/default.asp?query="),
      new SearchSpec("MSN",
                     "http://search.msn.com/results.asp?q="),
    };

  public static SearchSpec[] getCommonSpecs() {
    return(commonSpecs);
  }

  /** Given a search engine name and a search string, builds
   *  a URL for the results page of that search engine
   *  for that query. Returns null if the search engine name
   *  is not one of the ones it knows about.
   */

  public static String makeURL(String searchEngineName,
                               String searchString) {
    SearchSpec[] searchSpecs = getCommonSpecs();
    String searchURL = null;
    for(int i=0; i<searchSpecs.length; i++) {
      SearchSpec spec = searchSpecs[i];
      if (spec.getName().equalsIgnoreCase(searchEngineName)) {
        searchURL = spec.makeURL(searchString);
        break;
      }
    }
    return(searchURL);
  }
}
```

Listing 6.5 SearchEngineForm.java

```
package coreservlets;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

/** Servlet that builds the HTML form that gathers input
 *  for the search engine servlet. This servlet first
 *  displays a textfield for the search query, then looks up
 *  the search engine names known to SearchUtilities and
 *  displays a list of radio buttons, one for each search
 *  engine.
 */
```

```
public class SearchEngineForm extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
      throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String title = "One-Stop Web Search!";
    String actionURL = "/servlet/coreservlets.SearchEngines";
    String docType =
      "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 " +
      "Transitional//EN\">\n";
    out.println
      (docType +
       "<HTML>\n" +
       "<HEAD><TITLE>" + title + "</TITLE></HEAD>\n" +
       "<BODY BGCOLOR=\"#FDF5E6\">\n" +
       "<CENTER>\n" +
       "<H1>" + title + "</H1>\n" +
       "<FORM ACTION=\"" + actionURL + "\">\n" +
       "  Search keywords: \n" +
       "  <INPUT TYPE=\"TEXT\" NAME=\"searchString\"><P>\n");
    SearchSpec[] specs = SearchUtilities.getCommonSpecs();
    for(int i=0; i<specs.length; i++) {
      String searchEngineName = specs[i].getName();
      out.println("<INPUT TYPE=\"RADIO\" " +
                  "NAME=\"searchEngine\" " +
                  "VALUE=\"" + searchEngineName + "\">\n");
      out.println(searchEngineName + "<BR>\n");
    }
    out.println
      ("<BR>  <INPUT TYPE=\"SUBMIT\">\n" +
       "</FORM>\n" +
       "</CENTER></BODY></HTML>");
  }
}
```

Figure 6-3. Front end to the `SearchEngines` servlet. See Listing 6.5 for the source code.

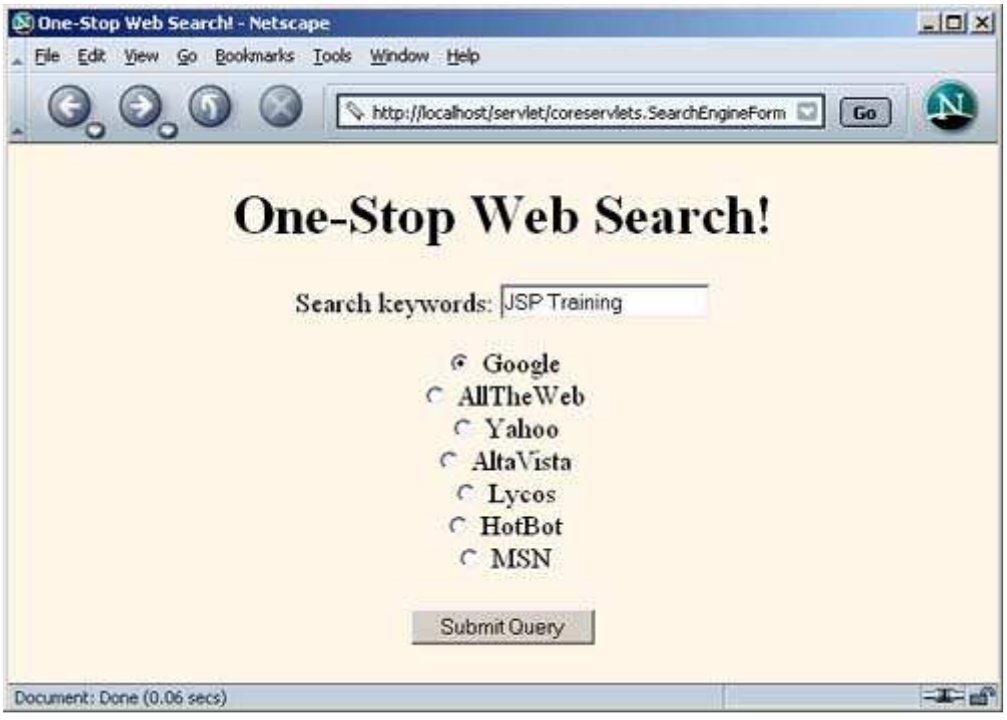


Figure 6-4. Results of the `SearchEngines` servlet when the form of Figure 6-3 is submitted. Although the form is submitted to the `SearchEngines` servlet, that servlet generates no output and the end user sees only the result of the redirection.

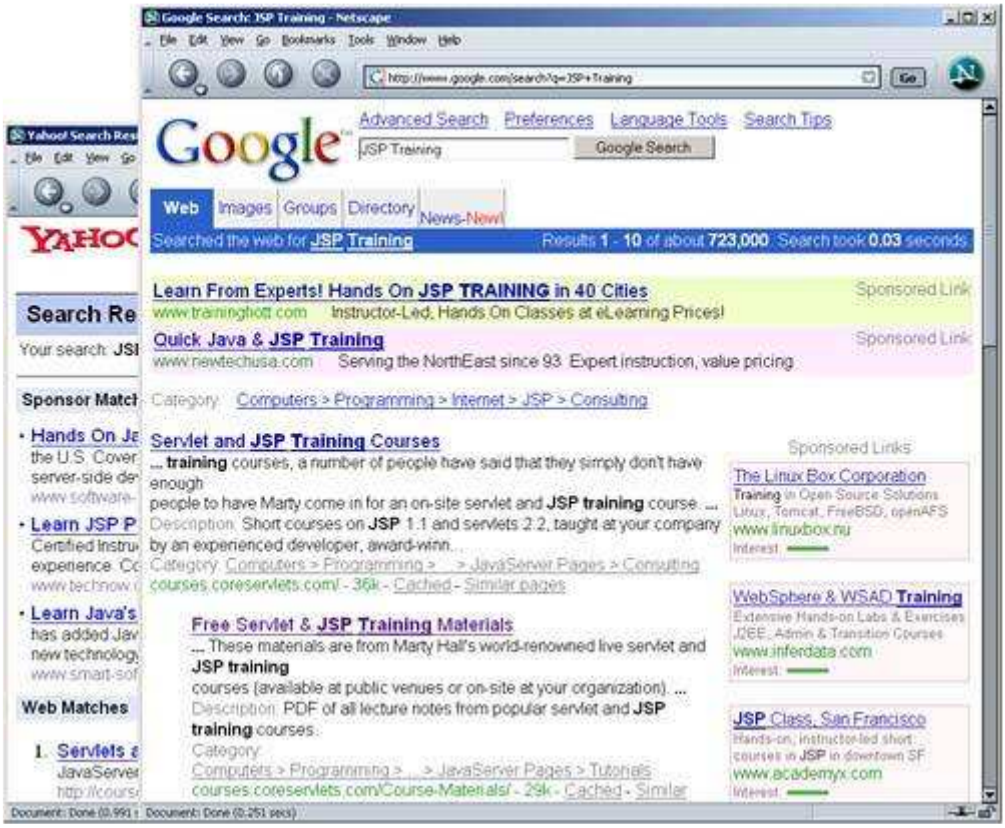
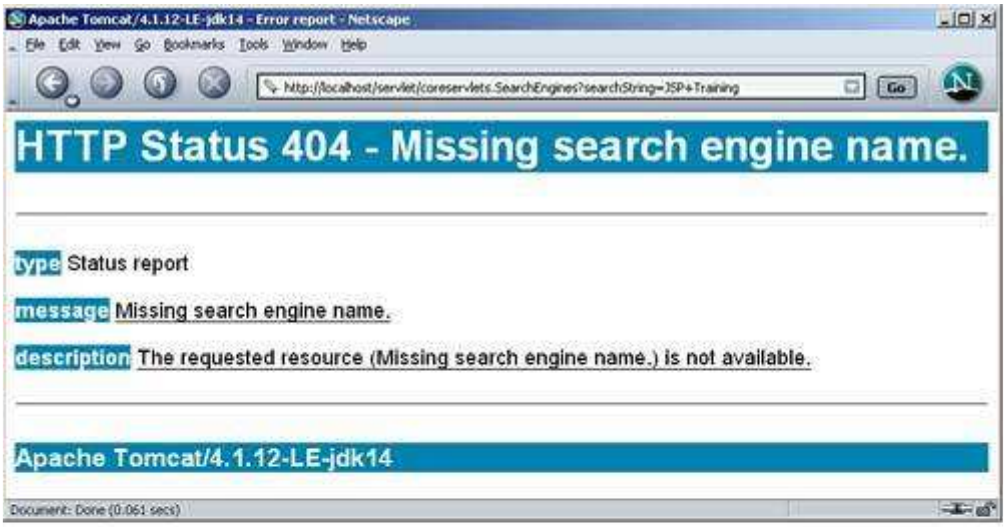
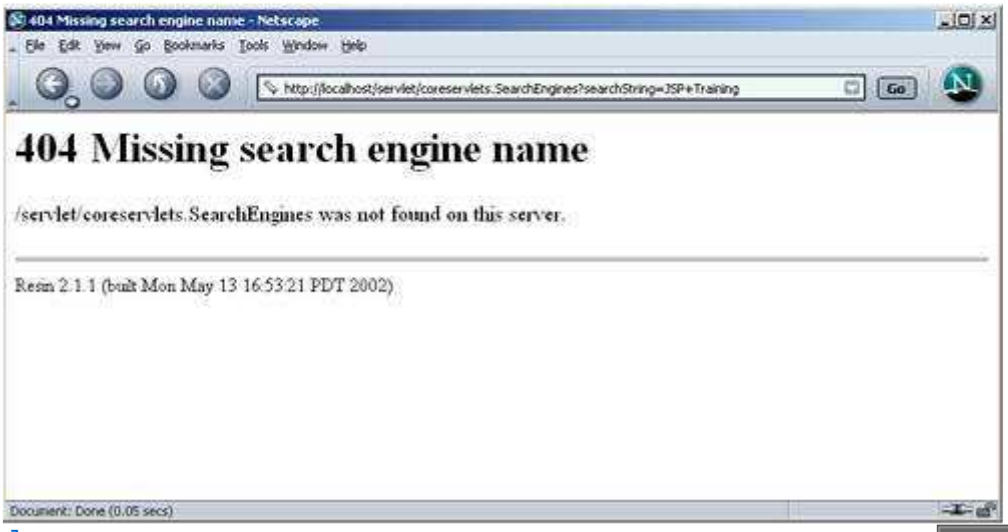


Figure 6-5. Results of the `SearchEngines` servlet upon submission of a form that has no search engine specified. These results are for Tomcat 4.1 and Resin 4.0; results will vary slightly among servers and will incorrectly omit the "Missing search string" message in JRun 4. In Internet Explorer, you must modify the browser settings as described in the previous section (see the 404 entry) to see the error message.


Apache Tomcat/4.1.12-LE-jdk14 - Error report - Netscape
File Edit View Go Bookmarks Tools Window Help
http://localhost/servlet/coreservlets.SearchEngines?searchString=JSP+Training
Go
HTTP Status 404 - Missing search engine name.
type Status report
message Missing search engine name.
description The requested resource (Missing search engine name.) is not available.
Apache Tomcat/4.1.12-LE-jdk14
Document: Done (0.061 secs)


404 Missing search engine name - Netscape
File Edit View Go Bookmarks Tools Window Help
http://localhost/servlet/coreservlets.SearchEngines?searchString=JSP+Training
Go
404 Missing search engine name
/servlet/coreservlets.SearchEngines was not found on this server.
Resin 2.1.1 (built Mon May 13 16:53:21 PDT 2002)
Document: Done (0.05 secs)