3.3 A Servlet That Generates HTML

Most servlets generate HTML, not plain text as in the previous example. To generate HTML, you add three steps to the process just shown:

1. Tell the browser that you're sending it HTML.
2. Modify the `println` statements to build a legal Web page.
3. Check your HTML with a formal syntax validator.

You accomplish the first step by setting the HTTP `Content-Type` response header to `text/html`. In general, headers are set by the `setHeader` method of `HttpServletResponse`, but setting the content type is such a common task that there is also a special `setContentType` method just for this purpose. The way to designate HTML is with a type of `text/html`, so the code would look like this:

```
response.setContentType("text/html");
```

Although HTML is the most common kind of document that servlets create, it is not unusual for servlets to create other document types. For example, it is quite common to use servlets to generate Excel spreadsheets (content type `application/vnd.ms-excel`—see Section 7.3), JPEG images (content type `image/jpeg`—see Section 7.5), and XML documents (content type `text/xml`). Also, you rarely use servlets to generate HTML pages that have relatively fixed formats (i.e., whose layout changes little for each request); JSP is usually more convenient in such a case. JSP is discussed in Part II of this book (starting in Chapter 10).

Don't be concerned if you are not yet familiar with HTTP response headers; they are discussed in Chapter 7. However, you should note now that you need to set response headers *before* actually returning any of the content with the `PrintWriter`. That's because an HTTP response consists of the status line, one or more headers, a blank line, and the actual document, *in that order*. The headers can appear in any order, and servlets buffer the headers and send them all at once, so it is legal to set the status code (part of the first line returned) even after setting headers. But servlets do not necessarily buffer the document itself, since users might want to see partial results for long pages. Servlet engines are permitted to partially buffer the output, but the size of the buffer is left unspecified. You can use the `getBufferSize` method of `HttpServletResponse` to determine the size, or you can use `setBufferSize` to specify it. You can set headers until the buffer fills up and is actually sent to the client. If you aren't sure whether the buffer has been sent, you can use the `isCommitted` method to check. Even so, the best approach is to simply put the `setContentType` line before any of the lines that use the `PrintWriter`.

Core Warning

*You must set the content type **before** transmitting the actual document.*

The second step in writing a servlet that builds an HTML document is to have your `println` statements output HTML, not plain text. Listing 3.3 shows `HelloServlet.java`, the sample servlet used in Section 2.8 to verify that the server is functioning properly. As Figure 3-3

illustrates, the browser formats the result as HTML, not as plain text.

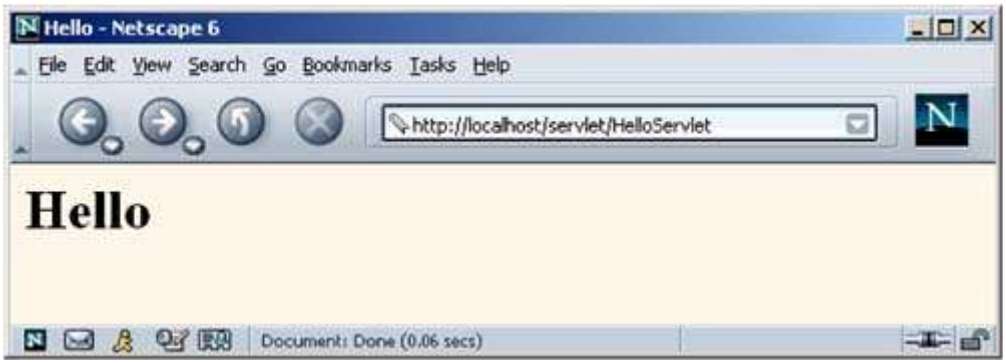
Listing 3.3 HelloServlet.java

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

/** Simple servlet used to test server. */

public class HelloServlet extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
      throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String docType =
      "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 " +
      "Transitional//EN\">\n";
    out.println(docType +
                "<HTML>\n" +
                "<HEAD><TITLE>Hello</TITLE></HEAD>\n" +
                "<BODY BGCOLOR=\"#FDF5E6\">\n" +
                "<H1>Hello</H1>\n" +
                "</BODY></HTML>");
  }
}
```

Figure 3-3. Result of `http://localhost/servlet/HelloServlet`.



The final step is to check that your HTML has no syntax errors that could cause unpredictable results on different browsers. See Section 3.5 (Simple HTML-Building Utilities) for a discussion of HTML validators.

[Team LiB]

PREVIOUS NEXT