CS360 Lecture 20
Servlets

Tuesday, April 27, 2004

Reading

Chapter 24

Servlets

Servlets are pieces of Java code that add functionality to a web server in a manner similar to the way applets add functionality to a browser. A servlet is a small, pluggable extension to a server that enhances the server's functionality. Servlets are most often used to extend web servers, providing a powerful, efficient replacement for CGI scripts.

Client-server relationship is a request/response model of communication. The servlets that we are interested in enhance the functionality of the World Wide Web servers.

Many developers feel that servlets are the right solution for database-intensive applications that communicate with thin clients.

Servlets vs. Java Server Pages

Servlets are used when only a small portion of the content sent to the client is static text or markup. Instead, they perform a task on behalf of the client. The opposite is true of JSPs.

Servlets and Java Server Pages are supported by the Apache web server, Microsoft's IIS, IBM's WebSphere and many more.

The servlets that we will look will demonstrate communication between clients and servers using the HTTP protocol.

Java Networking Packages

java.net for socket and packet based communications.
java.rmi packages for remote method invocation.
org.omg packages for CORBA.
javax.servlet and javax.servlet.http for servlets.
javax.servlet.jsp and javax.servlet.jsp.tagext for Java Server Pages.

HTTP Basics

HTTP is a simple, stateless protocol. When a client connects to a server and makes an HTTP request, the request can be of several different types, called methods, and the most frequently used are GET and POST.

The GET method is designed for getting information (a document or a result of a database query) and the POST method is designed for posting information (a credit card number, information to be stored in a database).

GET is the method used when you click on a hyperlink; either GET or POST can be used when submitting an HTML form.

Apache Tomcat

Tomcat is a Java servlet container and web server from the Jakarta project of the Apache Software Foundation. It is a free, open source servlet and JSP engine.

I have installed Tomcat on the natural science server. You can verify that it is up and running by typing in:
http://server.ns.pacificu.edu:8080/

I have also given you all permissions to startup and shutdown the server as necessary. I needed to do this because when you write your own servlets you will need to restart the server sometimes.

To do this, connect to server.ns.pacificu.edu using an ssh client and your username and password. Navigate to the directory: `Students/jakarta-tomcat-4.1.12-LE-jdk14/bin`

Once there, you can startup the server by typing `./startup.sh` and shutdown by typing `./shutdown.sh`

Directory Structure

Inside the folder `Students/jakarta-tomcat-4.1.12-LE-jdk14/webapps` you will find a folder with your user name. The example that I am using is created under the folder shereen. This is where you will place all your servlets and html files. Your folder should contain the following folders:

/servlets : This will contain all your html files.

/WEB-INF/classes : This will contain your Java class files. You could also store your Java source files here too, but it is not necessary.

/WEB-INF/web.xml : will list all your servlet information

Simple Servlet

The most basic type of servlet generates a full HTML page. The following servlet generates an html file containing the wording Hello World

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

```
public class WelcomeServlet extends HttpServlet {

   protected void doGet( HttpServletRequest request,
      HttpServletResponse response )
         throws ServletException, IOException
   {
      response.setContentType( "text/html" );
      PrintWriter out = response.getWriter();

      out.println( "<html>" );

      out.println( "<head>" );
      out.println( "<title>A Simple Servlet Example</title>" );
      out.println( "</head>" );

      out.println( "<body>" );
      out.println( "<h1>Hello World</h1>" );
      out.println( "</body>" );

      out.println( "</html>" );

      out.close();
   }
}
```

This file was compiled and placed in the shereen/WEB-INF/classes directory. You should compile it directly on the server as the server contains the javax.servlet packages. Again you can do this by connecting to the server using the ssh client.

HTML File

```
<html>
<head>
   <title>Handling an HTTP Get Request</title>
</head>

<body>
   <form action = "/shereen/welcome1" method = "get">

      <p><label>Click the button to invoke the servlet
         <input type = "submit" value = "Get HTML Document" />
      </label></p>
   </form>
</body>
</html>
```

This html file was placed in the shereen/servlets directory.

web.xml

```
<!DOCTYPE web-app PUBLIC
   "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
   "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">

<web-app>
   <!-- General description of your Web application -->
   <display-name>
      Java How to Program JSP and Servlet Chapter Examples
   </display-name>

   <description>
      This is the Web application in which we
      demonstrate our JSP and Servlet examples.
   </description>

   <!-- Servlet definitions -->
   <servlet>
      <servlet-name>welcome1</servlet-name>

      <description>
         A simple servlet that handles an HTTP get request.
      </description>

      <servlet-class>
         WelcomeServlet
      </servlet-class>
   </servlet>

   <!-- Servlet mappings -->
   <servlet-mapping>
      <servlet-name>welcome1</servlet-name>
      <url-pattern>/welcome1</url-pattern>
   </servlet-mapping>
</web-app>
```