Java Servlets and a Stand Alone Server

Stand Alone Server

A free stand alone server can be downloaded from the Apache open software project. It is jakarta-tomcat-5.5.7 and it is available from http://www.signal42.com/mirrors/apache/jakarta/tomcat-5/v5.5.7/bin/
You will also need Java version 5 to install it. You can download a current version at http://java.sun.com/j2se/1.5.0/download.jsp .

The installation of this version of tomcat is relatively easy. You should edit the file, context.xml, in the conf directory. Change the line <Context> to <Context reloadable="true">, so that the entire file is as follows:

```
<!-- The contents of this file will be loaded for each web application -->
<Context reloadable="true">

   <!-- Default set of monitored resources -->
      <WatchedResource>WEB-INF/web.xml</WatchedResource>
      <WatchedResource>META-INF/context.xml</WatchedResource>

      <!-- Uncomment this to disable session persistence across Tomcat restarts -->
      <!--
      <Manager pathname="" />
      -->
</Context>
```

You can leave the port that the server listens on at 8080, or you can change it to 80, the standard port for http. To change it, edit the file, server.xml, in the same conf directory. Find the number 8080 and change it to 80. It is in a section near the top that reads as follows:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
   <Connector port="8080"
            maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
            enableLookups="false" redirectPort="8443" acceptCount="100"
            connectionTimeout="20000" disableUploadTimeout="true" />
   <!-- Note : To disable connection timeouts, set connectionTimeout value to 0 -->
```

The first modification above will allow you to recompile servlets while the tomcat server is open. This is helpful when developing a project but unnecessary when the project is completed. The second change means that the 8080 can be dropped from the action line. The examples below assume that the port is 80.

The Action Attribute

There are a few more changes required for your html file as well. The main one is that the action attribute must include the term, servlet.

```
      action="http://localhost/servlet/client-server.EmailServlet/"
```

Note: This assumes that Tomcat has been modified to receive input on port 80. If you leave it on port 8080 (as initially configured), the action attribute must include the 8080.

```
      action="http://localhost:8080/servlet/client-server.EmailServlet/"
```

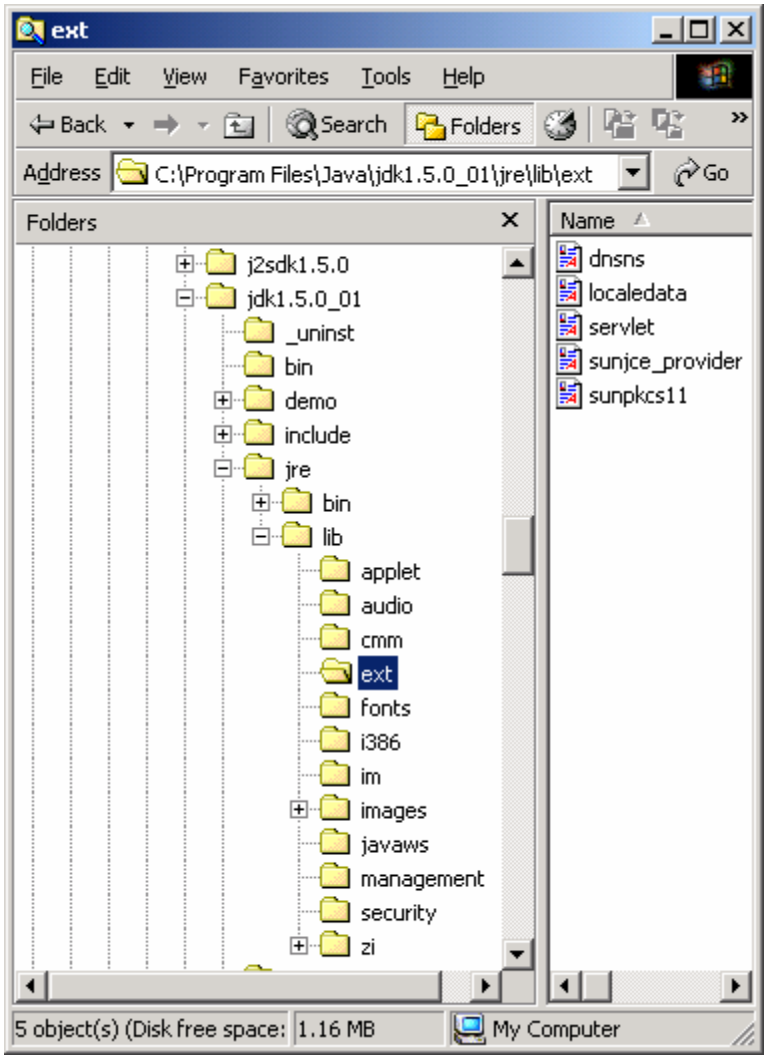
The rest of the form is the same as before.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
    <head><title>E-Mail Form</title></head>

<body>
    <h3>Enter your name and e-mail address.
    <br />Then click the Send button to send the data to the server.</h3>
    <form method = "get" action="http://localhost/servlet/client_server.EmailServlet">
        <p><input type = "text" name = "name" value = "" size = 30> Name </p>
        <p><input type = "text" name = "email" value = "" size = 30> E-Mail Address </p>
        <p><input type="submit" value="Send" /></p>
    </form>
</body> </html>
```

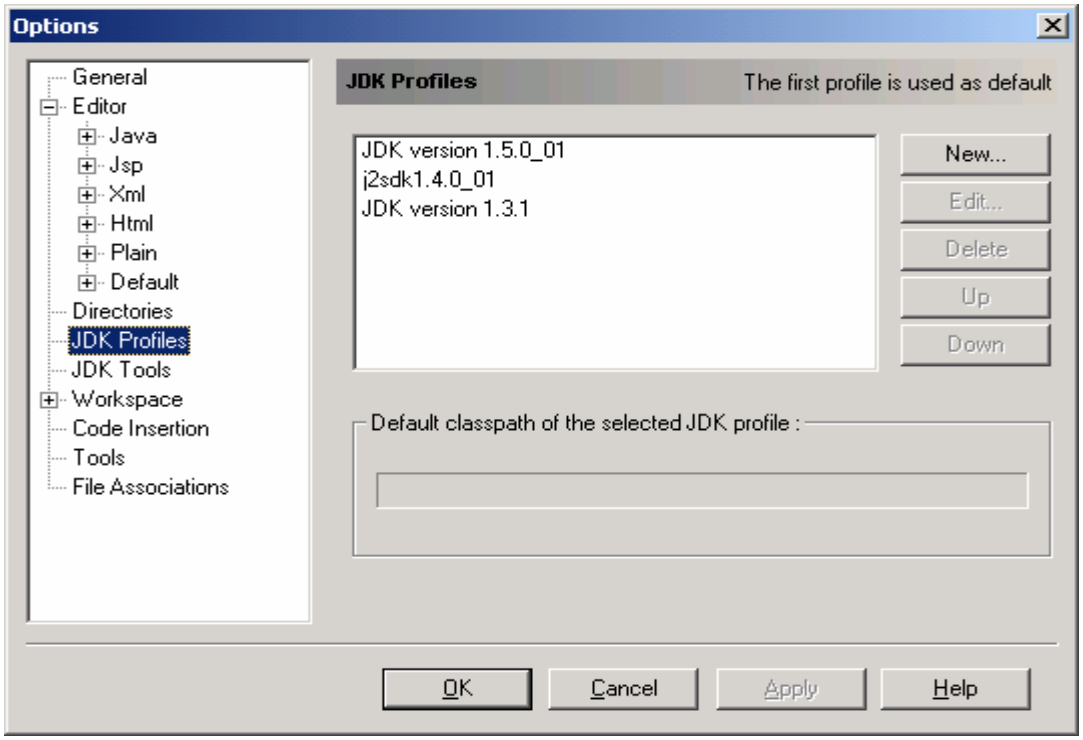
The servlet.jar package

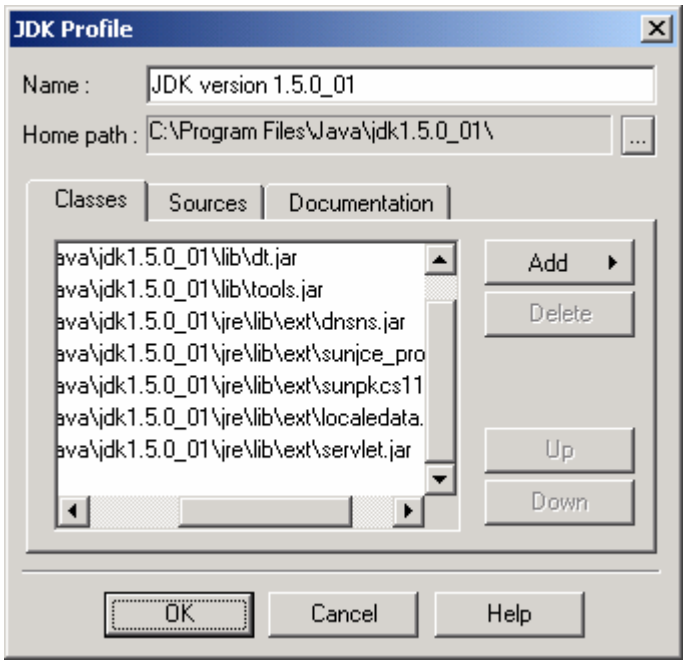


Unlike java.sql, javax.servlet does not come with the usual Java setup. It does come with Tomcat, but if you have problems downloading it, you will find a copy on my website. The package, servlet.jar, contains the java files needed for compiling Java servlets. The classes that are needed are javax.servlet and javax.servlet..http. They must both be imported into your file.

In addition, the Java compiler has to know where to locate these files. You can put the file anywhere, but the best place is in the folder c:\Program Files\Java\jdk1.5.0_01\jre\lib\ext. The server expects it to be there. And a lot of other jar files are there as well.

After that you will have to configure the classpath of your IDE (Integrated Development Environment) so that servlet.jar can be found when servlets are compiled. In JCreator, this is under Java Profiles. First click on **Configure/Options** and then on **JDK Profiles**. You should see the following screen.





Click on the compiler, here JDK version 1.5.0_01 and then **Edit**. Click on **Add/Add Package** and then navigate to the folder where you stored servlet.jar. When you click on servlet.jar, it will be added to the JCreator classpath. When done, it should look similar to the screen on the left.

For more details on how to set the server up, see the information at the following link.

http://www.coreservlets.com/Apache-Tomcat-Tutorial/

The Java Servlet

Only a few changes are required in order to modify the programs written for the WebServer. You have to import both javax.servlet and javax.servlet.http.

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

The processor class now must extend HttpServlet instead of WebRequestProcessor.

```
public class EmailServlet extends HttpServlet
```

The main method in the servlet is now either *doGet* or *doPost* and the parameters are HttpServletRequest and HttpServletResponse.

```
public void doGet (HttpServletRequest request, HttpServletResponse response)
```

Finally this method always throws an IOException that must be caught or re-thrown.

he EmailServlet Example

```
package client_server;

/* EmailServlet processes a request from a web page.  It responds to the request by echoing back the
name and email address that was sent in. */
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class EmailServlet extends HttpServlet
{
	String name, email;
	PrintWriter out;

	public void doGet (HttpServletRequest request, HttpServletResponse response)
	{
		try
		{
			out = response.getWriter ();
			name = request.getParameter ("name");
			email = request.getParameter ("email");

			// Get a PrintWriter object and respond to the request.
			//PrintWriter out = response.getWriter ();

			// The Page class is contained in the client_server folder.
			Page.createHeader (out, "Test Data");
			out.println ("<h3>Hello.</h3>");
			out.println ("<h3>" + name+ "</h3>");
			out.println ("<h3>Your email address is " + email + "</h3>");
			Page.createFooter (out);
		}catch (IOException e) {System.out.println ("Servlet Exception");}
	} // doGet
} // EmailServlet
```