References

Servlet Essentials, http://www.novocode.com/doc/servlet-essentials/

Axis documentation and examples, http://ws.apache.org/axis/

Tomcat documentation and examples, http://jakarta.apache.org/

SOAP Server Installing

Things needed

- Ant – Java based build tool

http://ant.apache.org/

Needed to build Axis

- Tomcat – servlet container and Server

http://jakarta.apache.org/

Needed to act as web server for Axis

Install before Axis as it has the basic servlet classes used by Axis

- Axis – Apache SOAP

http://ws.apache.org/axis/

Tomcat

Download from:

http://jakarta.apache.org/tomcat/index.html

Use version 4.1.18.

For documentation see:

http://jakarta.apache.org/tomcat/tomcat-4.1-doc/index.html

For instructions on installing and running see:

http://jakarta.apache.org/tomcat/tomcat-4.1-doc/RUNNING.txt

Running Tomcat

In the Tomcat installation go to the bin directory

On Windows run:

startup

On Unix machines run:

startup.sh

By default Tomcat runs on port 8080

Use web browser to connect to Tomcat

On the same machine use:

http://localhost:8080/

If successful the browser will show the default Tomcat home page

Servlets

Mechanism to handle http requests via Java

Servlets Verses CGI

A Servlet

- Does not run in a separate process from the Web Server
- Stays in memory between requests
- Uses one instance to handle all requests concurrently

Example – Hello World

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

public class HelloWorld extends HttpServlet {
   public void doGet(HttpServletRequest request,
      HttpServletResponse response)
      throws IOException, ServletException
   {
      response.setContentType(" text/html ");
      PrintWriter out = response.getWriter();
      out.println(" <html> ");
      out.println(" <body> ");
      out.println(" <head> ");
      out.println(" <title>Hello World!</title> ");
      out.println(" </head> ");
      out.println(" <body> ");
      out.println(" <h1>Hello World!</h1> ");
      out.println(" </body> ");
      out.println(" </html> ");
   }
}
```

Some Servlet Basics

HttpServlet Methods

doGet (HttpServletRequest req, HttpServletResponse resp)
doPost
doPut
doDelete

Methods called when http get, post, put, delete requests are sent to servlet url

init(ServletConfig config)

Container stores ServletConfig for later use by servlet

public ServletConfig **getServletConfig** ()

Returns stored ServletConfig

An Example – Hi Mom

Java Source

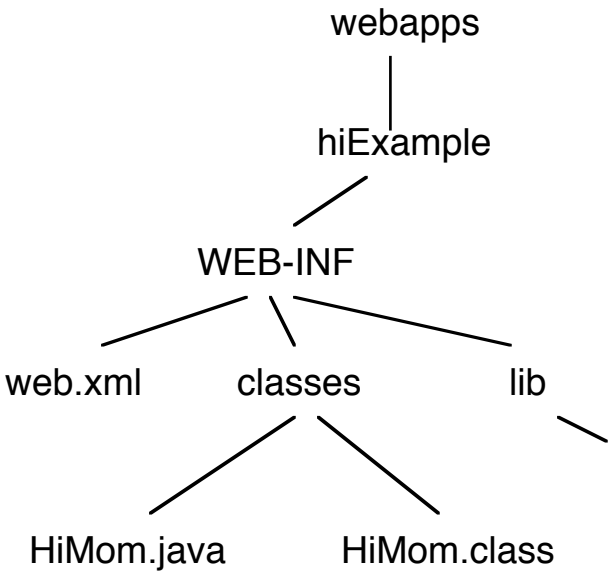
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HiMom extends HttpServlet {
   public void doGet(HttpServletRequest request,
      HttpServletResponse response)
      throws IOException, ServletException
   {
      response.setContentType(" text/html ");
      PrintWriter out = response.getWriter();
      out.println(" <html> ");
      out.println(" <body> ");
      out.println(" <head> ");
      out.println(" <title>Hi Mom </title> ");
      out.println(" </head> ");
      out.println(" <body> ");
      out.println(" <h1>Hi Mom </h1> ");
      out.println(" </body> ");
      out.println(" </html> ");
   }
}
```

Installing, Compiling & Running Example

Tomcat installation contains directory webapps

Structure needed



hiExample – directory containing web application (one servlet)

web.xml – information about servlets in application

See http://jakarta.apache.org/tomcat/tomcat-4.1-doc/appdev/web.xml.txt for an example

Sample web.xml File

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

   <display-name>My First Servlet Example</display-name>
   <description>
   Just shows how to install and run a Servlet with Tomcat
   </description>

   <servlet>
     <servlet-name>mom</servlet-name>
     <description>
     </description>
     <servlet-class>HiMom</servlet-class>
   </servlet>

   <servlet-mapping>
     <servlet-name>mom</servlet-name>
     <url-pattern>hi</url-pattern>
   </servlet-mapping>

   <session-config>
   <session-timeout>30</session-timeout>
   </session-config>
</web-app>
```

Servlet Showing Request Data

```
public class ShowRequest extends HttpServlet {
      public void doGet(HttpServletRequest request,
            HttpServletResponse response)
            throws IOException, ServletException
   {
      response.setContentType(" text/html ");
      PrintWriter out = response.getWriter();
      out.println(" <html> ");
      out.println(" <body> ");
      out.println("<table border=0><tr><td>");
      out.println("requestinfo.label.method");
      out.println("</td><td>");
      out.println(request.getMethod());
      out.println("</td></tr><tr><td>");
      out.println("requestinfo.label.requesturi");
      out.println("</td><td>");
      out.println(request.getRequestURI());
      out.println("</td></tr><tr><td>");
      out.println("requestinfo.label.protocol");
      out.println("</td><td>");
      out.println(request.getProtocol());
      out.println("</td></tr><tr><td>");
      out.println("requestinfo.label.pathinfo");
      out.println("</td><td>");
      out.println(request.getPathInfo());
      out.println("</td></tr><tr><td>");
      out.println("requestinfo.label.remoteaddr");
      out.println("</td><td>");
      out.println(request.getRemoteAddr());
      out.println("</table>");
      out.println(" </body> ");
      out.println(" </html> ");
   }}
```

Ant

Download at:
http://ant.apache.org/bindownload.cgi

See the manual at http://ant.apache.org/manual/index.html

The command:

```
ant
```

will

- Read the local file build.xml
- Run commanded in the build.xml file

Axis

Download from
http://ws.apache.org/axis/

You need to build the axis.jar file using ant

Make sure that the servlet.jar from Tomcat install is in your classpath when building Axis

Optional jars (also in Tomcat install)

activation.jar
http://java.sun.com/products/javabeans/glasgow/jaf.html

mailapi.jar
http://java.sun.com/products/javamail/

After unpacking Axis go to the Axis/java directory and type:

ant

When successfully build copy axis/java/webapps contents to your Tomcat webapps directory

Make sure that the jar files in axis/java/build/lib are in your classpath

Restart Tomcat

With a browser go to http://localhost:8080/axis/index.html

You will get the standard Axis Web page

Follow the Validate link to insure that Axis is working

Sample Soap Service

Example from Axis distribution and

Calculator.jws

```
public class Calculator {
  public int add(int i1, int i2)
  {
    return i1 + i2;
  }


  public int subtract(int i1, int i2)
  {
    return i1 - i2;
  }
}
```

Place the Java file in the Tomcat/webapps.axis directory

Axis will compile etc for you.

Sample Client

Based on Client in Axis Distribution

```
import org.apache.axis.client.Call;
import org.apache.axis.client.Service;
import org.apache.axis.encoding.XMLType;
import org.apache.axis.utils.Options;

import javax.xml.rpc.ParameterMode;

public class Client {
   public static void main(String [] args) throws Exception {

       Integer i1 = new Integer(2);
       Integer i2 = new Integer(3);
       String endpoint = "http://localhost:8080/axis/Calculator.jws";

       Service  service = new Service();
       Call     call    = (Call) service.createCall();

       call.setTargetEndpointAddress( new java.net.URL(endpoint) );
       call.setOperationName( "add" );
       call.addParameter( "op1", XMLType.XSD_INT, ParameterMode.IN );
       call.addParameter( "op2", XMLType.XSD_INT, ParameterMode.IN );
       call.setReturnType( XMLType.XSD_INT );

       Integer ret = (Integer) call.invoke( new Object [] { i1, i2 });

       System.out.println("Got result : " + ret);
   }
}
```