SECTION 1 ADVANCED INTERNET TECHNOLOGIES – LAB

Structure **Page Nos.**

1.0 INTRODUCTION

In this section of lab course, you will have hands on experience of Advanced Internet Technologies, which includes programming practices in Java Servlet and Java Server Pages (JSP), web page security, and XML. This section is based on course MCS–051: Advanced Internet Technologies. A list of programming exercises (Lab Assignments) is given at the end of this section. You will have to write programs to solve these exercises and execute it during lab sessions.

Java Servlet technology and Java Server Pages (JSP) are server-side technologies. They are now becoming the standard way to develop commercial web applications. Java Technologies basic feature Servlet JSP bring the "Write Once, Run Anywhere" paradigm to web applications. If, you apply effectively best practices, servlets and JSP to help separate presentation from content, this will help in the understanding applications and reduces complexity.

Before you begin programming/attempt programming, it is essential to learn how container/web servers are to be installed to handle Servlet/JSP applications. In this section, first you will learn about Jakarta Tomcat installation, then, about the execution of Servlet and JSP programs.

The objective of XML design is to describe data and at the same time to focus on what data is. You may not realise, but all that the XML does to create structures, store and to send information. Now, it has started becoming evident that XML will be as important to the future of Web Applications as HTML has been to the foundation of Web Development. In this section, you will learn to create and view XML documents.

1.1 OBJECTIVES

After completing this lab section, you should be able to:

- Install Jakarta Tomcat;
- Set CLASSPATH;
- Change Server Port;
- Develop and deploy Servlet programs;
- Develop and deploy JSP programs;
- Write secure Servlets, and
- Create XML documents.

1.2 TOMCAT INSTALLATIONS AND SETTING

Tomcat is the servlet container that is used in official Reference Implementation for Java Servlet and Java Server Pages Technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process.

There are some other web servers/ servlet container, Jetty is one of them. It is a 100% Java HTTP Server and Servlet Container. This means that you do not need to configure and run a separate web server (like, Apache which you will also learn about in this unit) in order to use java, servlets and JSPs to generate dynamic content. The Jetty web server can be used for static and dynamic content. As a pure Java component, Jetty can be included in your application for demonstration, and deployment. You can learn more about Jetty at:

1. http://www.java-source.net/open-source/web-servers

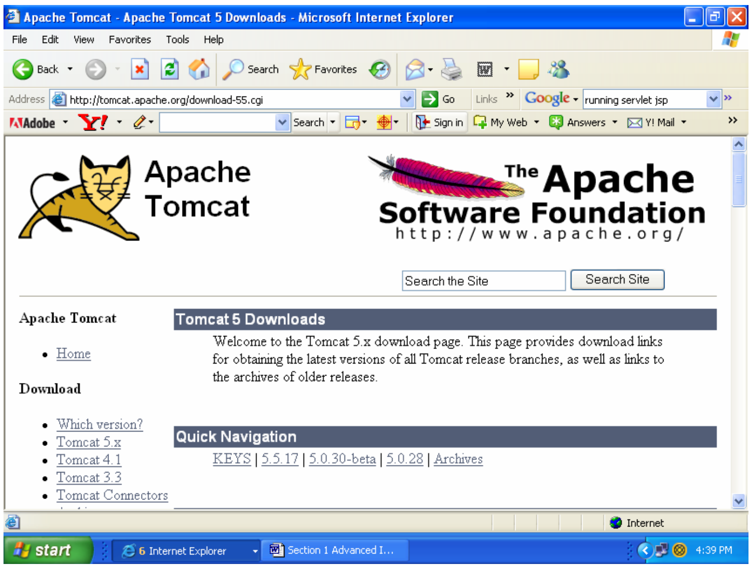
2. http://www.roseindia.net/opensource/webserversinjava.php

Downloading and Installing Tomcat:

We will use Jakarta Tomcat for our servlet and JSP programming. Now, let us see, how tomcat is to be installed on your machine.

Do the following:

1) Go to http://tomcat.apache.org/download-55.cgi as shown in *Figure 1* and download the zip file for the current release of Tomcat (Jakarta Tomcat 5.5)



Apache Tomcat

- Home

Download

- Which version?
- Tomcat 5.x
- Tomcat 4.1
- Tomcat 3.3
- Tomcat Connectors

Tomcat 5 Downloads

Welcome to the Tomcat 5.x download page. This page provides download links for obtaining the latest versions of all Tomcat release branches, as well as links to the archives of older releases.

Quick Navigation

KEYS | 5.5.17 | 5.0.30-beta | 5.0.28 | Archives

Figure 1: Tomcat homepage

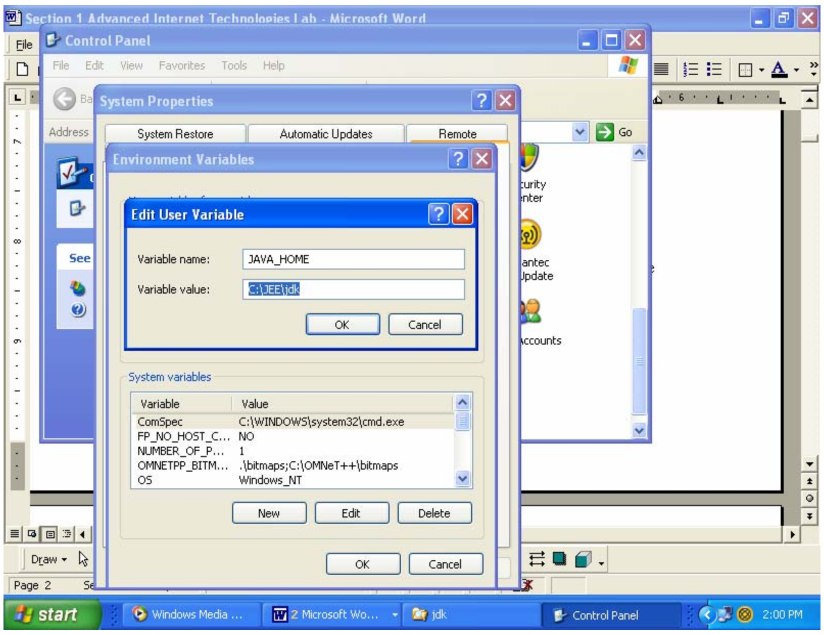
2) Save the zip file(s) on your desktop and unzip it.

3) You would need to specify the directory (e.g., *C:\Tomcat5.5) so that* C:*\Tomcat5.5\apache-tomcat-5.5.15 becomes a* common resultant installation directory for you.

Set the JAVA_HOME Variable

Now, you will have to set JAVA_HOME environment variable to inform Tomcat where Java is located. If, you do not set this variable properly then, Tomcat will not compile JSP pages. This variable should list the base JDK installation directory, not the *bin* subdirectory. For example, on Windows, if you installed the JDK in *C:\JEE\jdk* you might put the following line in your setting of JAVA_HOME.
set JAVA_HOME= C:\JEE\jdk

If you are working on n Windows XP, you could also go to the Start menu> select Control Panel>choose System, click on the Advanced tab, press the Environment Variables button at the bottom, and enter the JAVA_HOME variable and value directly as shown in *Figure 2*. Similarly, for other OS environments setting can be done.



Variable name: JAVA_HOME
Variable value: C:\JEE\jdk

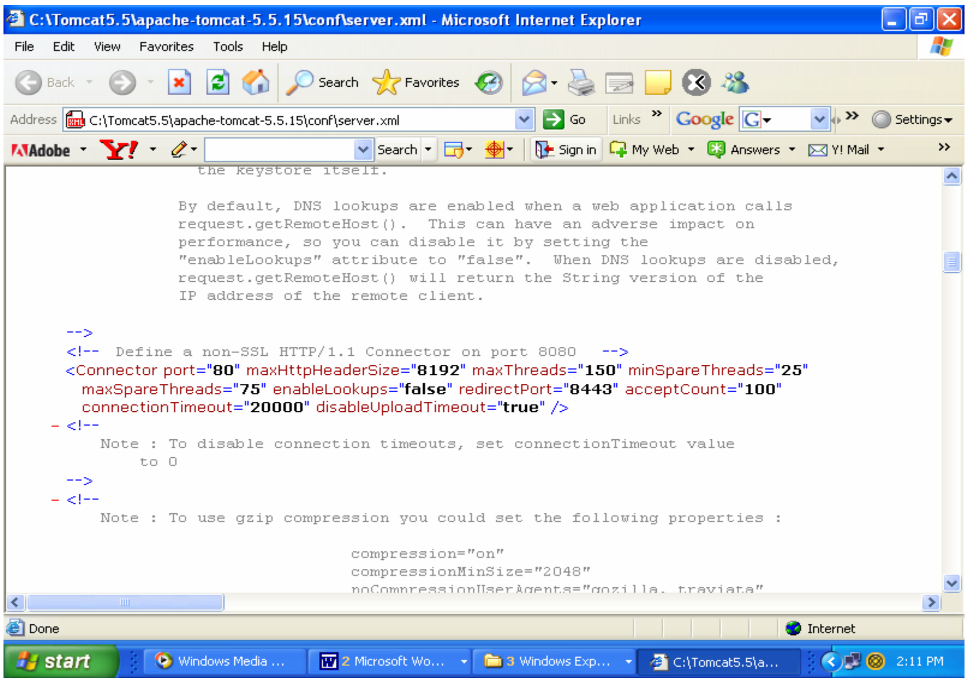
Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	1
OMNETPP_BITM...	.\bitmaps;C:\OMNeT++\bitmaps
OS	Windows_NT

Figure 2: Setting JAVA_HOME

Change the Port to 80

You will have to change the port to 80 for Tomcat to run on the default HTTP port (80) instead of the port of 8080.I assume that you have no other server already running on port 80. It will be convenient to configure it, and making this change will let you use URLs of the form *http://localhost/urex* instead of *http://localhost:8080/urex*.On Unix/Linux you need admin privileges to make this change. If, you are using IIS, it will automatically start on port 80. If, you want to use port 80 for Tomcat, you may need to disable IIS.

To change the port, go to C:\Tomcat5.5\apache-tomcat-5.5.15\conf\server.xml and change the port attribute of the Connector element from 8080 to 80, as shown in *Figure 3.*



```
    the keystore itself.

         By default, DNS lookups are enabled when a web application calls
         request.getRemoteHost().  This can have an adverse impact on
         performance, so you can disable it by setting the
         "enableLookups" attribute to "false".  When DNS lookups are disabled,
         request.getRemoteHost() will return the String version of the
         IP address of the remote client.

   -->
   <!--  Define a non-SSL HTTP/1.1 Connector on port 8080   -->
   <Connector port="80" maxHttpHeaderSize="8192" 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
   -->
 - <!--
       Note : To use gzip compression you could set the following properties :

                           compression="on"
                           compressionMinSize="2048"
                           noCompressionUserAgents="gozilla, traviata"
```

C:\Tomcat5.5\apache-tomcat-5.5.15\conf\server.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Tomcat5.5\apache-tomcat-5.5.15\conf\server.xml
Done
Internet
start
2:11 PM

Figure 3: Modifying Server.xml

Verify Server Start

Now, your server is ready for service. Before trying your own servlets or JSP programs you should check that the server is installed and configured properly. Two batch files start*up.bat* and shutdoun.bat are given in C:\Tomcat5.5\apache-tomcat-5.5.15\bin. It is better to have shortcut of files start*up.bat* and shutdoun.bat files on your desktop.

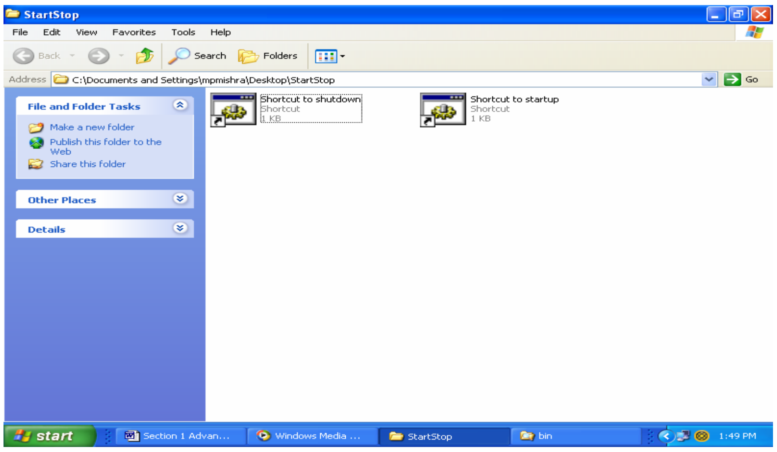
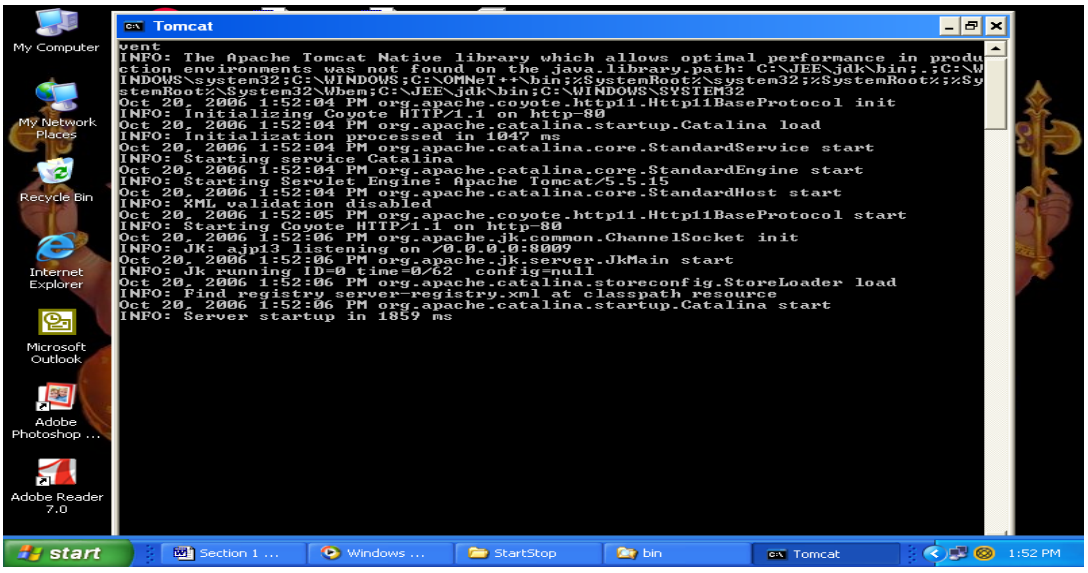


Figure 4: Startup.bat and Shutdoun.bat Files on a Folder on Desktop

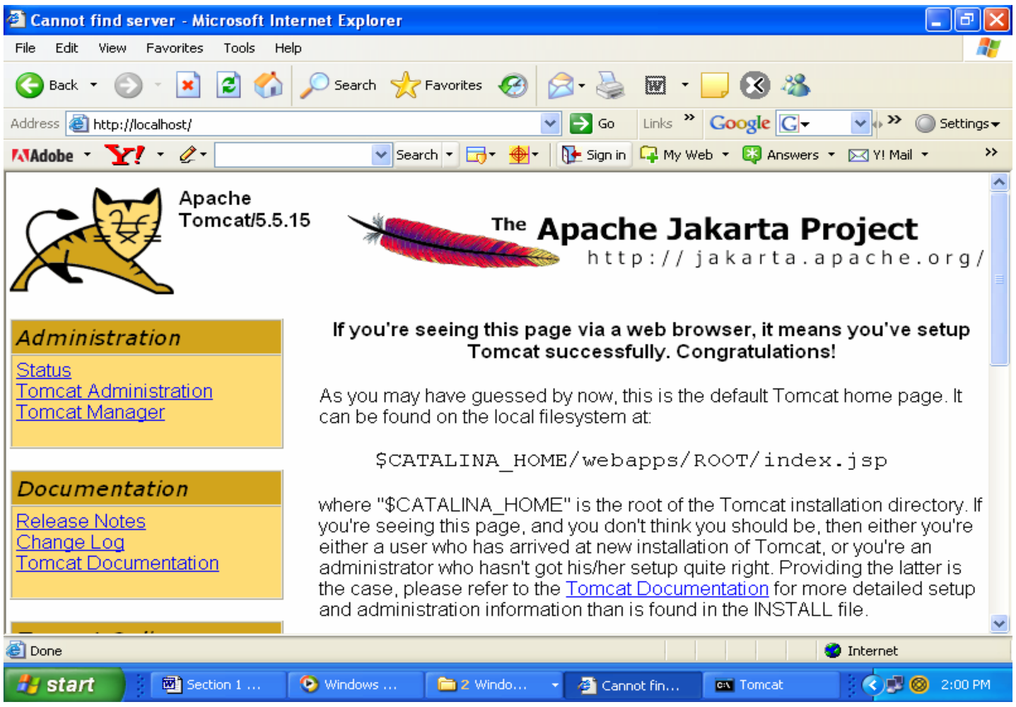
Now, start your server by double-clicking on shortcut to startup. You will find window shown in *Figure 5.*



```
vent
INFO: The Apache Tomcat Native library which allows optimal performance in produ
ction environments was not found on the java.library.path: C:\JEE\jdk\bin;.;C:\W
INDOWS\system32;C:\WINDOWS;C:\OMNeT++\bin;%SystemRoot%\system32;%SystemRoot%;%Sy
stemRoot%\System32\Wbem;C:\JEE\jdk\bin;C:\WINDOWS\SYSTEM32
Oct 20, 2006 1:52:04 PM org.apache.coyote.http11.Http11BaseProtocol init
INFO: Initializing Coyote HTTP/1.1 on http-80
Oct 20, 2006 1:52:04 PM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 1047 ms
Oct 20, 2006 1:52:04 PM org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
Oct 20, 2006 1:52:04 PM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/5.5.15
Oct 20, 2006 1:52:04 PM org.apache.catalina.core.StandardHost start
INFO: XML validation disabled
Oct 20, 2006 1:52:05 PM org.apache.coyote.http11.Http11BaseProtocol start
INFO: Starting Coyote HTTP/1.1 on http-80
Oct 20, 2006 1:52:06 PM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Oct 20, 2006 1:52:06 PM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/62  config=null
Oct 20, 2006 1:52:06 PM org.apache.catalina.storeconfig.StoreLoader load
INFO: Find registry server-registry.xml at classpath resource
Oct 20, 2006 1:52:06 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 1859 ms
```

Figure 5: Running Tomcat

Now, your server is started. The next step is to, enter the URL *http://localhost/* in your browser and see whether you reach/are able to access the Tomcat welcome page or not. Welcome page is shown in *Figure 6.*



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

`$CATALINA_HOME/webapps/ROOT/index.jsp`

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then either you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

Figure 6: Default Tomcat Homepage

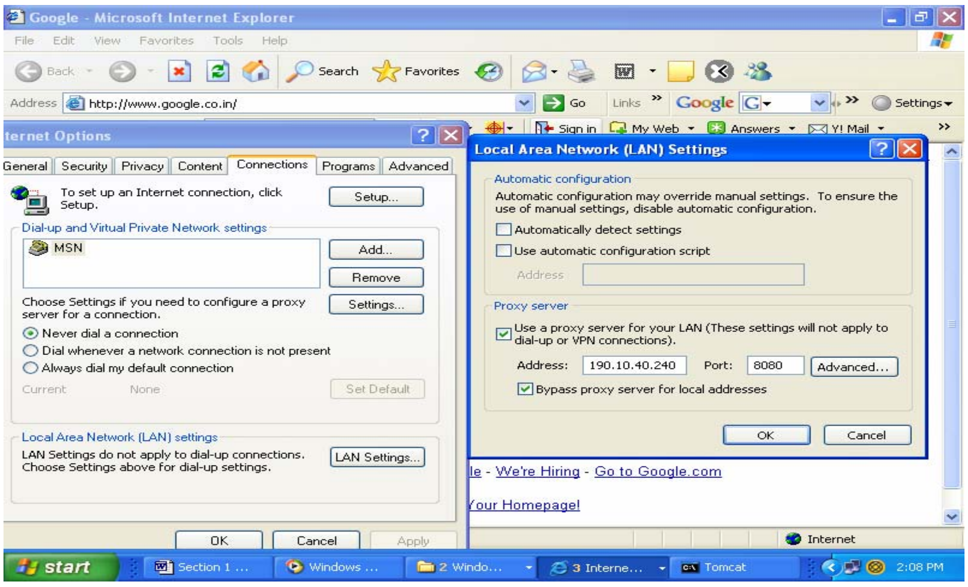
If, you get an error message saying that the page could not be displayed or that the server could not be found then, there is some problem in your installation and setting.

If this does not work, there are a couple of things to check:

- Go through the error messages; it should help you find out the problem (e.g., JAVA_HOME not set properly or IIS already reserving port 80).

- Check whether the server appears to be running but you cannot access the home page? There is a chance that your browser is using a proxy and you have not set it to bypass proxies for local addresses. If this is the case then:

In Internet Explorer, go to Tools>>Internet Options>>Connections>> LAN Settings. Here, if the "Use a proxy server" checkbox is selected, make sure the "Bypass proxy server for local addresses" box is also selected, as shown in *Figure 7.*


Local Area Network (LAN) Settings

Automatic configuration

Automatic configuration may override manual settings. To ensure the use of manual settings, disable automatic configuration.

☐ Automatically detect settings

☐ Use automatic configuration script

Proxy server

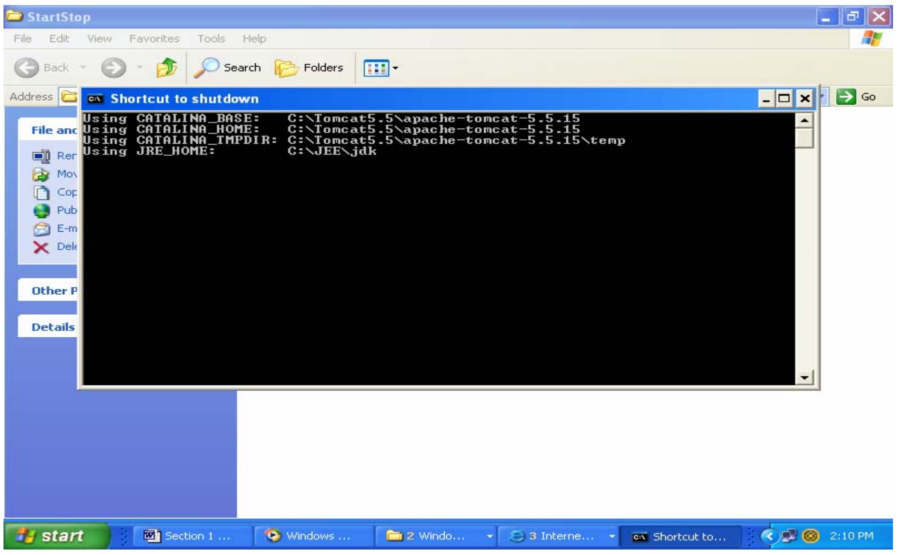
☑ Use a proxy server for your LAN (These settings will not apply to dial-up or VPN connections).

Address: 190.10.40.240 Port: 8080

☑ Bypass proxy server for local addresses

Figure 7: Setting to Bypass Proxy Server for Local Addresses

To halt the server, double click shortcut of *shutdown.bat* :


Shortcut to shutdown

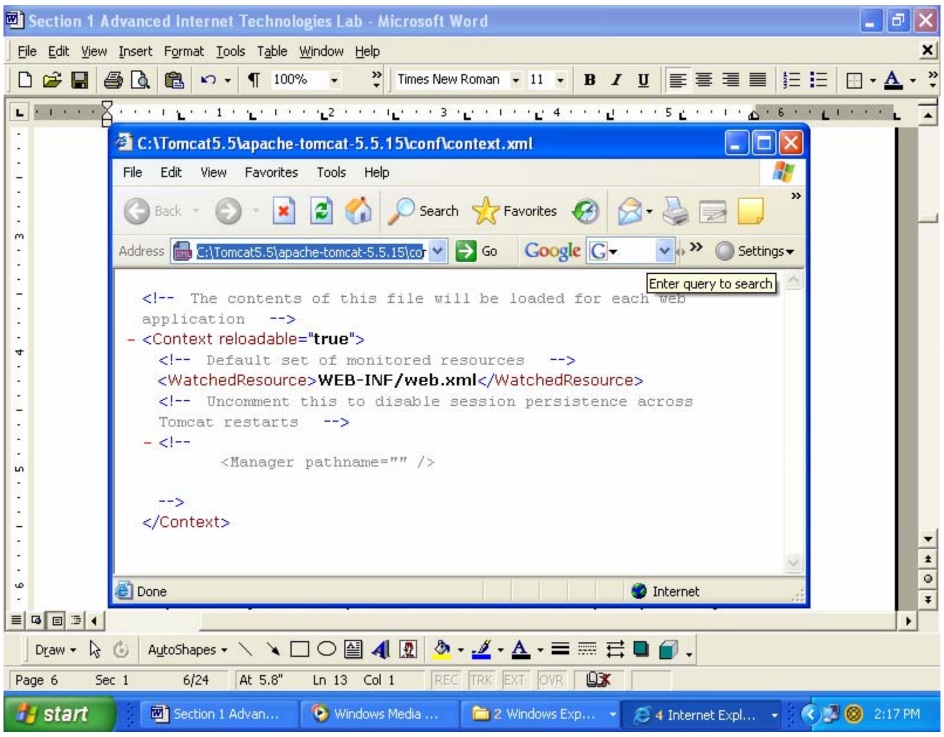
```
Using CATALINA_BASE:   C:\Tomcat5.5\apache-tomcat-5.5.15
Using CATALINA_HOME:   C:\Tomcat5.5\apache-tomcat-5.5.15
Using CATALINA_TMPDIR: C:\Tomcat5.5\apache-tomcat-5.5.15\temp
Using JRE_HOME:        C:\JEE\jdk
```

Figure 8: Shutdown Tomcat

Turn on Servlet Reloading

To run your servlet you have to put its class file in to WEB-*INF/classes* and use the URL http://localhost/servlet/ServletName or http://localhost/webAppName/servlet/ServletName) Here in http://localhost/webAppName/servlet/ServletName, webApp-Name is the name of your subdirectory in WEB-INF/classes directory. If, the class files of your servlet programs get modified it will be seen by Tomcat. You have to set it by doing the following:

***<Context reloadable="true">*, in** *C:\Tomcat5.5\apache-tomcat-5.5.15\conf\context.xml as shown in Figure 9.*



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

    -->
  </Context>
```

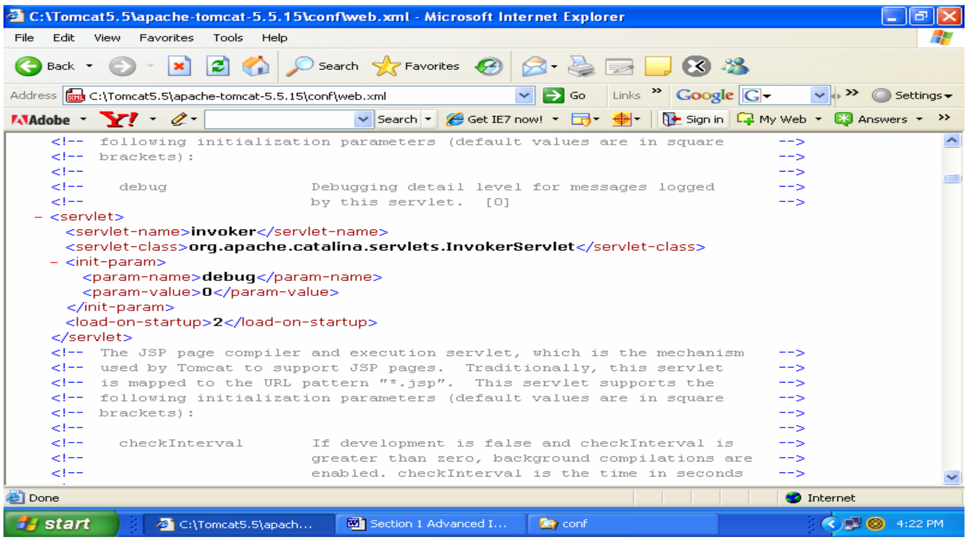
Figure 9: Modifying Context.xml

Tomcat check the modification dates of the class files of requested servlets, and reload ones that have changed since they were loaded into the server's memory. If you don't turn reloadable on for your development server, you have to restart the server every time you recompile a servlet, which is already loaded into the server's memory.

Enabling the Invoker Servlet

As a beginner, you need to run your initial servlet programs without making any change in your Web application's deployment descriptor file (in *WEB-INF/web.xml* file). For this, you will have to get invoker servlet enabled. After this, you just drop your servlet into
WEB-INF/classes and use the URL *http://host/servlet/ServletName* (or *http://host/webAppName /servlet/ServletName)* once you start using your own Web applications.
You have to, uncomment the following servlet and servlet-mapping elements in *install_dir/conf/web.xml* as shown in *Figure 10* and *Figure 11*, for enabling the invoker servlet.Remember this is not Apache Tomcat-specific *web.xml* file which goes in the *WEB-INF* directory of each Web application.

1.
```
<servlet>
        <servlet-name>invoker</servlet-name>
        <servlet-class>
          org.apache.catalina.servlets.InvokerServlet
        </servlet-class>
        ...
    </servlet>
```

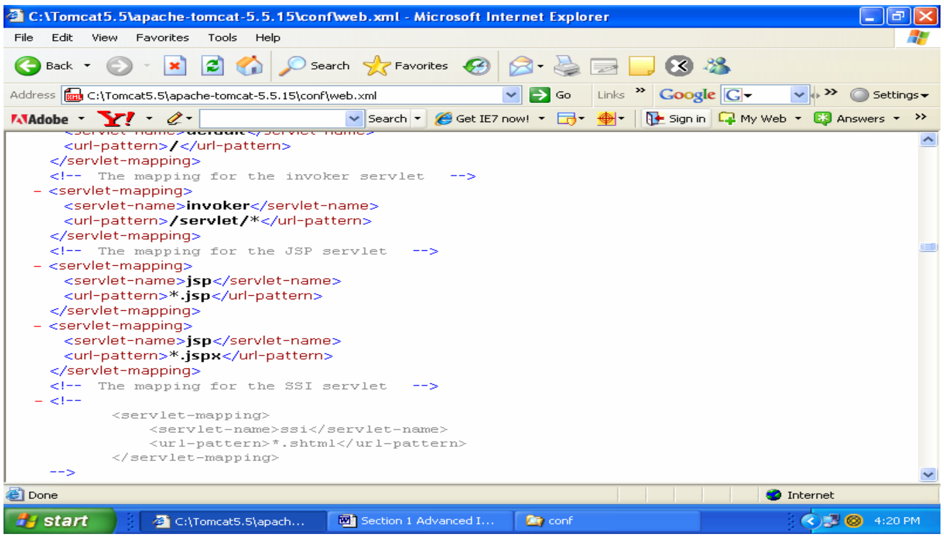


C:\Tomcat5.5\apache-tomcat-5.5.15\conf\web.xml - Microsoft Internet Explorer

```
   <!--  following initialization parameters (default values are in square      -->
   <!--  brackets):                                                              -->
   <!--                                                                          -->
   <!--    debug                  Debugging detail level for messages logged     -->
   <!--                           by this servlet.   [0]                         -->
 - <servlet>
     <servlet-name>invoker</servlet-name>
     <servlet-class>org.apache.catalina.servlets.InvokerServlet</servlet-class>
   - <init-param>
       <param-name>debug</param-name>
       <param-value>0</param-value>
     </init-param>
     <load-on-startup>2</load-on-startup>
   </servlet>
   <!--  The JSP page compiler and execution servlet, which is the mechanism     -->
   <!--  used by Tomcat to support JSP pages.  Traditionally, this servlet       -->
   <!--  is mapped to the URL pattern "*.jsp".  This servlet supports the        -->
   <!--  following initialization parameters (default values are in square      -->
   <!--  brackets):                                                              -->
   <!--                                                                          -->
   <!--    checkInterval          If development is false and checkInterval is   -->
   <!--                           greater than zero, background compilations are -->
   <!--                           enabled. checkInterval is the time in seconds  -->
```

Figure 10: Modifying Web.xml for Enabling the Invoker Servlet

2.
```
<servlet-mapping>
    <servlet-name>invoker</servlet-name>
    <url-pattern>/servlet/*</url-pattern>
</servlet-mapping>
```



C:\Tomcat5.5\apache-tomcat-5.5.15\conf\web.xml - Microsoft Internet Explorer

```
     <url-pattern>/</url-pattern>
   </servlet-mapping>
   <!--  The mapping for the invoker servlet   -->
 - <servlet-mapping>
     <servlet-name>invoker</servlet-name>
     <url-pattern>/servlet/*</url-pattern>
   </servlet-mapping>
   <!--  The mapping for the JSP servlet   -->
 - <servlet-mapping>
     <servlet-name>jsp</servlet-name>
     <url-pattern>*.jsp</url-pattern>
   </servlet-mapping>
 - <servlet-mapping>
     <servlet-name>jsp</servlet-name>
     <url-pattern>*.jspx</url-pattern>
   </servlet-mapping>
   <!--  The mapping for the SSI servlet   -->
 - <!--
         <servlet-mapping>
             <servlet-name>ssi</servlet-name>
             <url-pattern>*.shtml</url-pattern>
         </servlet-mapping>
   -->
```

Figure 11: Uncommenting Servlet-mapping in Web.xml file

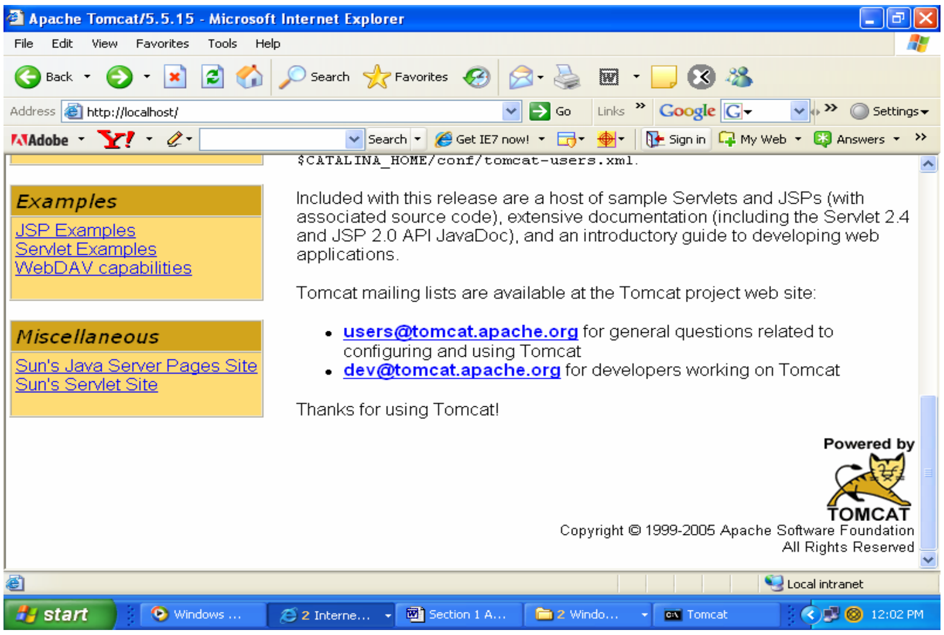
Set CLASSPATH

Servlets and JSP are not part of the Java 2 platform, standard edition that is why the *compiler* (i.e., javac) you use for development does not know about Servlet and JSP, and you will have to identify the servlet classes to the compiler. So, you have to set your CLASSPATH properly, and attempt to compile servlets, tag libraries, Web app listeners, or other classes that use the servlet and JSP, otherwise, APIs will fail with error messages saying unknown classes. You need to include both files in your CLASSPATH for standard Tomcat Server:

- C:\Tomcat5.5\apache-tomcat-5.5.15\common\lib\servlet-api.jar, and
- C:\Tomcat5.5\apache-tomcat-5.5.15\common\lib\jsp-api.jar

If, you have created your own development directory then you also need to put that directory in the CLASSPATH. It will be good (learning purpose), for you to write simple package less servlets. After you gain enough experience you have to use packages for your applications development.

Now, you are at the stage where you can test your own html, servlet, and jsp programs in Tomcat Server. But, before you test your own programs, I will suggest the execution of some examples of Servlet and JSP page given by Tomcat.


Apache Tomcat/5.5.15 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/ Go Links Google Settings
Adobe Search Get IE7 now! Sign in My Web Answers
Local intranet
start Windows ... 2 Interne... Section 1 A... 2 Windo... Tomcat 12:02 PM

$CATALINA_HOME/conf/tomcat-users.xml.

Examples
JSP Examples
Servlet Examples
WebDAV capabilities

Miscellaneous
Sun's Java Server Pages Site
Sun's Servlet Site

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation (including the Servlet 2.4 and JSP 2.0 API JavaDoc), and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- users@tomcat.apache.org for general questions related to configuring and using Tomcat
- dev@tomcat.apache.org for developers working on Tomcat

Thanks for using Tomcat!

Powered by TOMCAT

Copyright © 1999-2005 Apache Software Foundation
All Rights Reserved

Figure 12: Link to Servlet and JSP Example on Tomcat Default Page

Now you can try Some Simple HTML and JSP Page

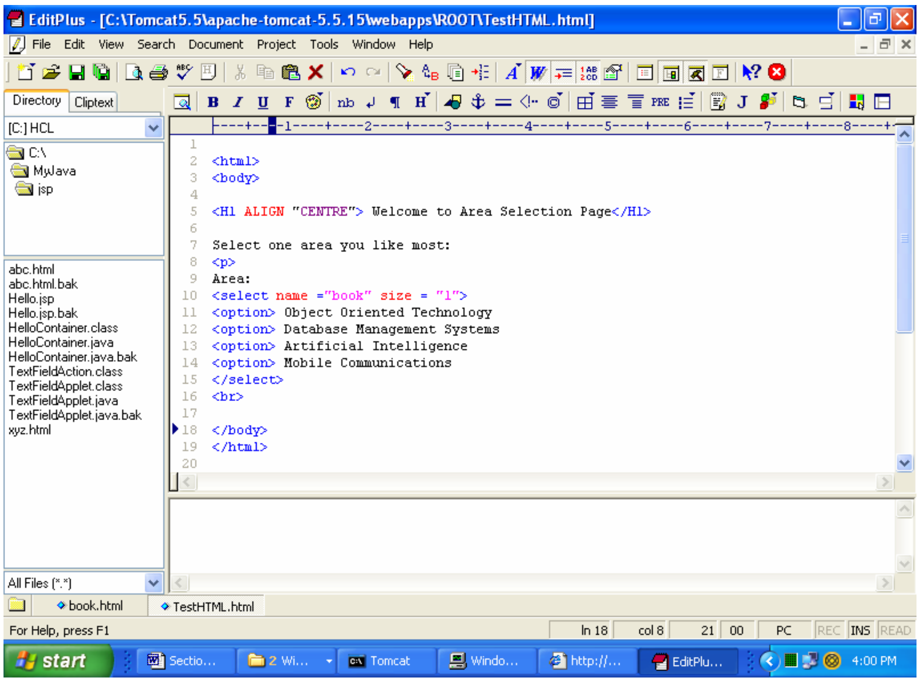
First, verify that the server is running. If, the server is running, you have to make sure that you can install and access simple HTML and JSP:

- If, you can access an HTML page successfully, it will help you ensure that HTML and JSP files are loaded in the right/correct directories, and the URLs correspond make sure for you that in which directories,

- If, you are able to successfully access a new JSP page then, it shows that you have configured the Java *compiler* properly.

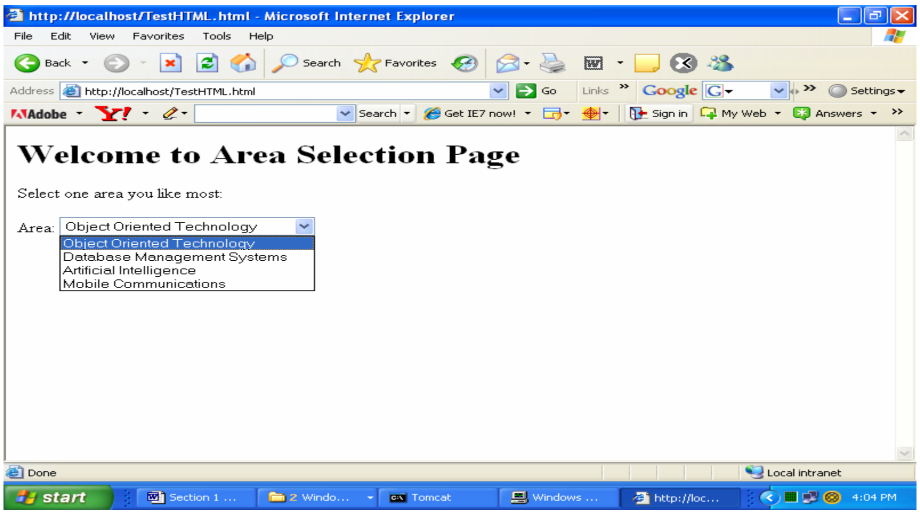
To run in Tomcat, HTML and JSP pages need to be put in install_dir/webapps/ROOT or install_dir/webapps/ROOT/somePath and need to be accessed with http://localhost/filename or http://localhost/somePath/filename. In our case we have to put HTML and JSP files in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT directory or a subdirectory in this directory, created by you.

For example TestHTML.html file is saved in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT directory and when you access it with URL http://localhost/TestHTML.html you will find the page given in *Figure 14.*



```
<html>
<body>

<H1 ALIGN "CENTRE"> Welcome to Area Selection Page</H1>

Select one area you like most:
<p>
Area:
<select name ="book" size = "1">
<option> Object Oriented Technology
<option> Database Management Systems
<option> Artificial Intelligence
<option> Mobile Communications
</select>
<br>

</body>
</html>
```

Figure 13: A simple HTML file



Welcome to Area Selection Page

Select one area you like most:

Area: Object Oriented Technology

Object Oriented Technology
Database Management Systems
Artificial Intelligence
Mobile Communications

Figure 14: Viewing TestHTML.html from a Local Host

1.3 RUNNING JSP PROGRAMS

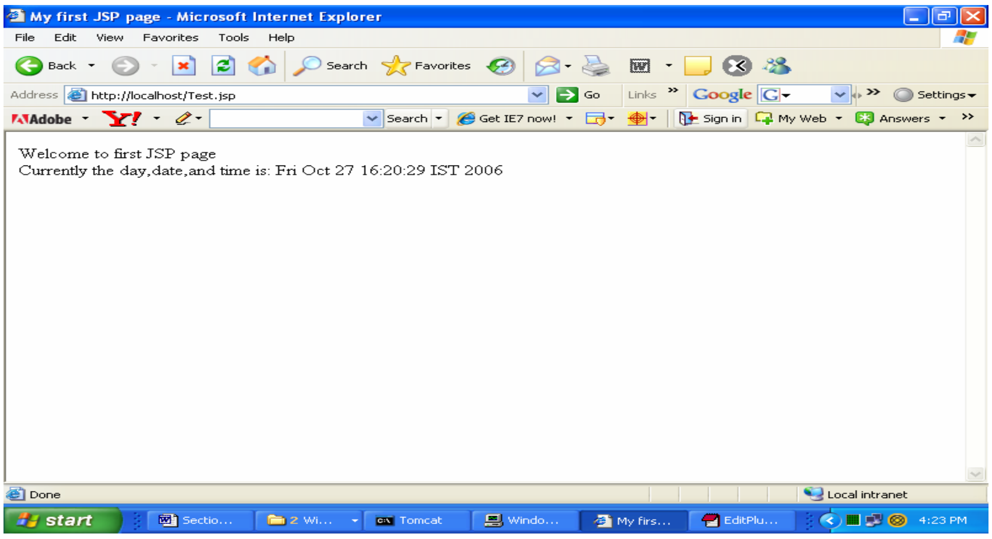
Now, let us take a simple JSP program saved in file Test.jsp which contains the following code to show current date and time:

JSP File – Test.jsp

```
<html>

<head>

<title>My first JSP page

</title>
```

```
</head>

<body>

<%    // This scriptlet declares and initialises "date"

      java.util.Date date = new java.util.Date();

%>

 <% out.println("Welcome to first JSP page"); %>

 <BR>

  <%

    out.println("Currently the day,date,and time is");

    out.println( date );

%>

</body>

</html>
```

Save this Test.jsp file in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT directory and access it with URL http://localhost/Test.jsp, you will get Web Page shown in *Figure 15a*.



```
Welcome to first JSP page
Currently the day,date,and time is: Fri Oct 27 16:20:29 IST 2006
```

My first JSP page - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/Test.jsp
Go
Done
Local intranet
start
4:23 PM

Figure 15 a: Output of Test.jsp

Now, let us see an example in which a java class is used for counting the number of times this JSP is accessed.

Java Class File - Counter.java

```
public class  Counter
```

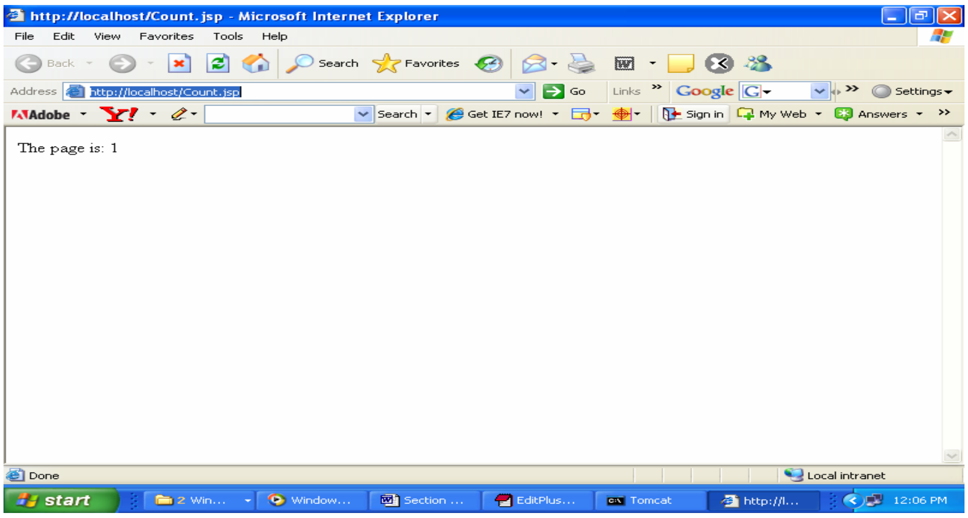
```
{
        private static int counter;
        public static int getCounter()
        {
                counter++;
                return counter;
        }
}
```

JSP File – Cont.jsp

```
<html>
<body>
The page is:
<% out.println(Counter.getCounter());
%>
</body>
</html>
```

You have to compile Counter.java file and save Counter.class file in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes directory.

Save Count.jsp in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT directory.Start server and access URL : http://localhost/Count.jsp trough Internet Explorer , you will get following screen :


http://localhost/Count.jsp - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost/Count.jsp
Done
Local intranet

```
The page is: 1
```

Figure 15 b: Output of Count.jsp

The number of times you will access through URL : : http://localhost/Count.jsp will increase by one The page is: 1, The page is: 2 etc.

1.4 RUNNING SERVLET PROGRAMS

Now, let us test the execution of a servlet program. As you know, the servlet generates an HTML page and for its development two things are needed—servlet API class files which are used for servlet compilation and web server for deployment of servlet. In addition to J2SE you would also need to download and install Tomcat, which has the servlet API (which is in servlet-api.jar file) the compiler is searching for. As you know, your servlet classes import the javax.servlet and javax.servlet.http package.

You have to make sure that those packages are accessible by your compiler, those, which, you have already added in your CLASSPATH as the servlet-api.jar file.

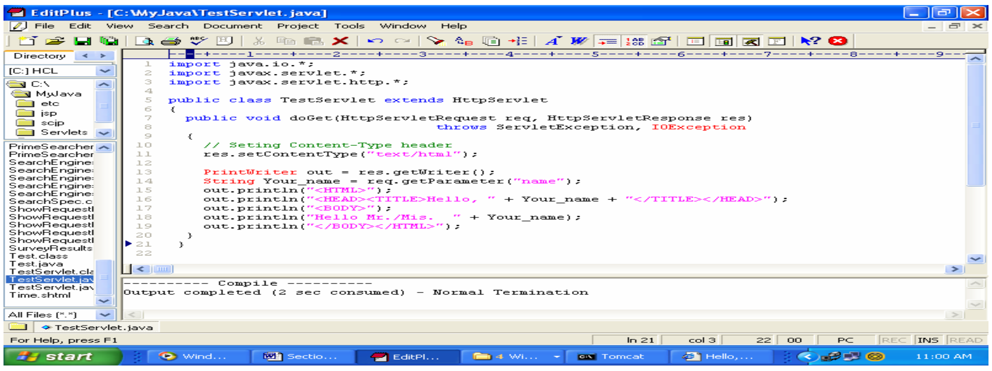
Now, you can compile your servlet program with the javac compiler that comes with the JSDK.You have to store.class file in the deployment directory manually or you have to use appropriate compilation option –d<directory>.

In Tomcat, the location for servlets in the default Web application is *install_dir/webapps/ROOT/WEB-INF/classes*. In our case of installation, it will be, C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes directory, you have to create *WEB-INF* and *classes* directory yourself (Remember that text case matters in the name of these directories.).

Now, you are ready to write servlet programs as you have written Java programs in course MCSL025.Compile it and copy *.class* file in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes directory, as shown in *Figure 17*.

Now, compile TestServlet.java servlet program given below:

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

public class TestServlet extends HttpServlet
{
 public void doGet(HttpServletRequest req, HttpServletResponse res)
                              throws ServletException, IOException
  {
   // Seting Content-Type header
   res.setContentType("text/html");

   PrintWriter out = res.getWriter();
   String Your_name = req.getParameter("name");
   out.println("<HTML>");
   out.println("<HEAD><TITLE>Hello, " + Your_name + "</TITLE></HEAD>");
   out.println("<BODY>");
   out.println("Hello Mr./Mis.  " + Your_name);
   out.println("</BODY></HTML>");
  }
 }
```

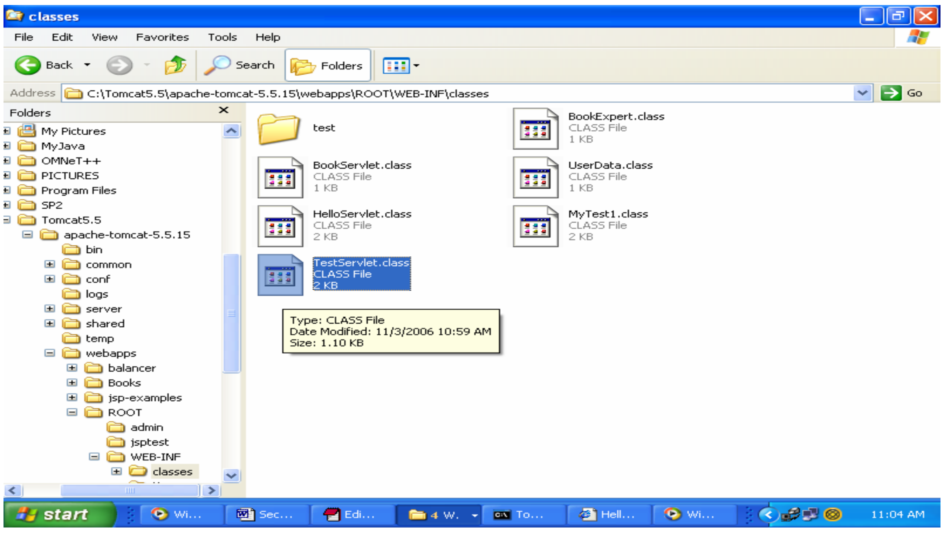


```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class TestServlet extends HttpServlet
{
  public void doGet(HttpServletRequest req, HttpServletResponse res)
                                throws ServletException, IOException
  {
    // Seting Content-Type header
    res.setContentType("text/html");

    PrintWriter out = res.getWriter();
    String Your_name = req.getParameter("name");
    out.println("<HTML>");
    out.println("<HEAD><TITLE>Hello, " + Your_name + "</TITLE></HEAD>");
    out.println("<BODY>");
    out.println("Hello Mr./Mis.   " + Your_name);
    out.println("</BODY></HTML>");
  }
 }
```

```
---------- Compile ----------
Output completed (2 sec consumed) - Normal Termination
```

Figure 16: TestServlet .java in EditPlus

Copy TestServlet.class file in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes directory.



Address: C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes

test
BookExpert.class CLASS File 1 KB
BookServlet.class CLASS File 1 KB
UserData.class CLASS File 1 KB
HelloServlet.class CLASS File 2 KB
MyTest1.class CLASS File 2 KB
TestServlet.class CLASS File 2 KB

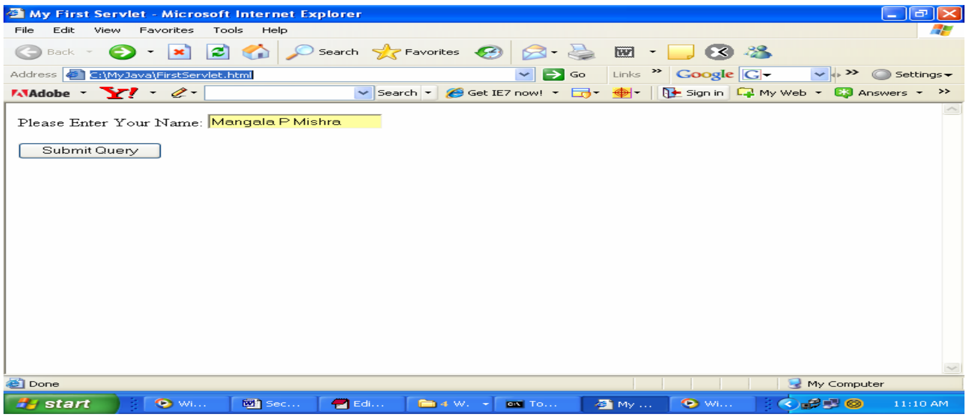
Type: CLASS File
Date Modified: 11/3/2006 10:59 AM
Size: 1.10 KB

Figure 17: Copying TestServlet.class file in C directory

Write following HTML code:

```
<HTML>

<HEAD><TITLE>My First Servlet</TITLE></HEAD>

<BODY>

<FORM METHOD = GET ACTION= "http://localhost/servlet/TestServlet">

  Please Enter Your Name:

  <INPUT  TYPE = TEXT NAME = "name"><p>

  <INPUT TYPE = SUBMIT>

  </FORM>

</BODY></HTML>
```

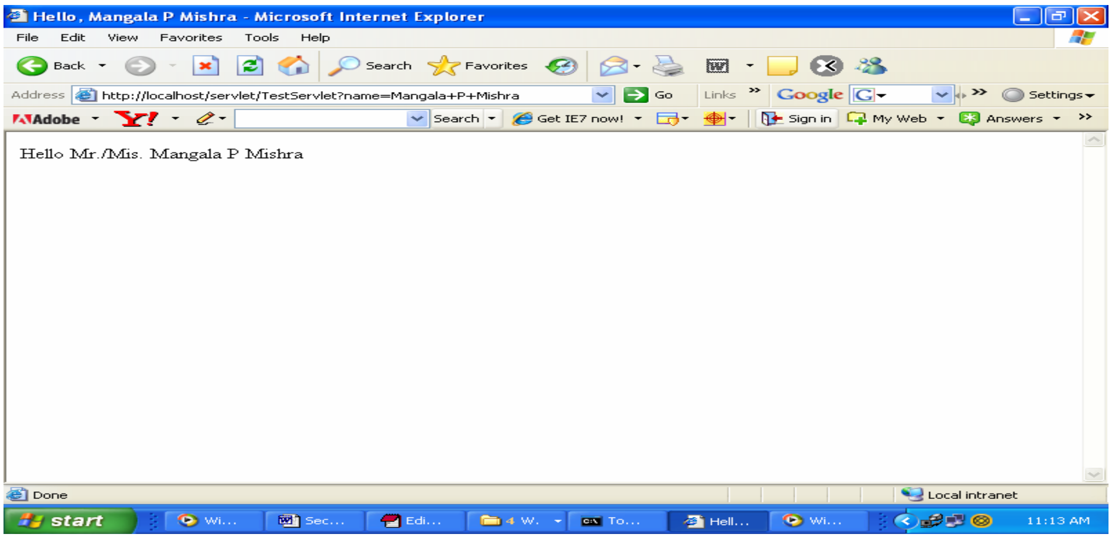
Save this HTML code in FirstServlet.html file. Now, you have an HTML code for invoking TesrServlet program. Start your Tomcat and invoke the URL C:\MyJava\FirstServlet.html. You will find following the form as the output asking for your name. Enter a name into textfield and press Submit Query button, you will go to URL http://localhost/servlet/TestServlet and you will find web page shown in *Figure 19a.*


My First Servlet - Microsoft Internet Explorer
C:\MyJava\FirstServlet.html
Submit Query
Done
My Computer

Please Enter Your Name: Mangala P Mishra

Figure 18: FirstServlet.html page

This is the URL for your servlet program TestServlet.



http://localhost/servlet/TestServlet?name=Mangala+P+Mishra

Hello Mr./Mis. Mangala P Mishra

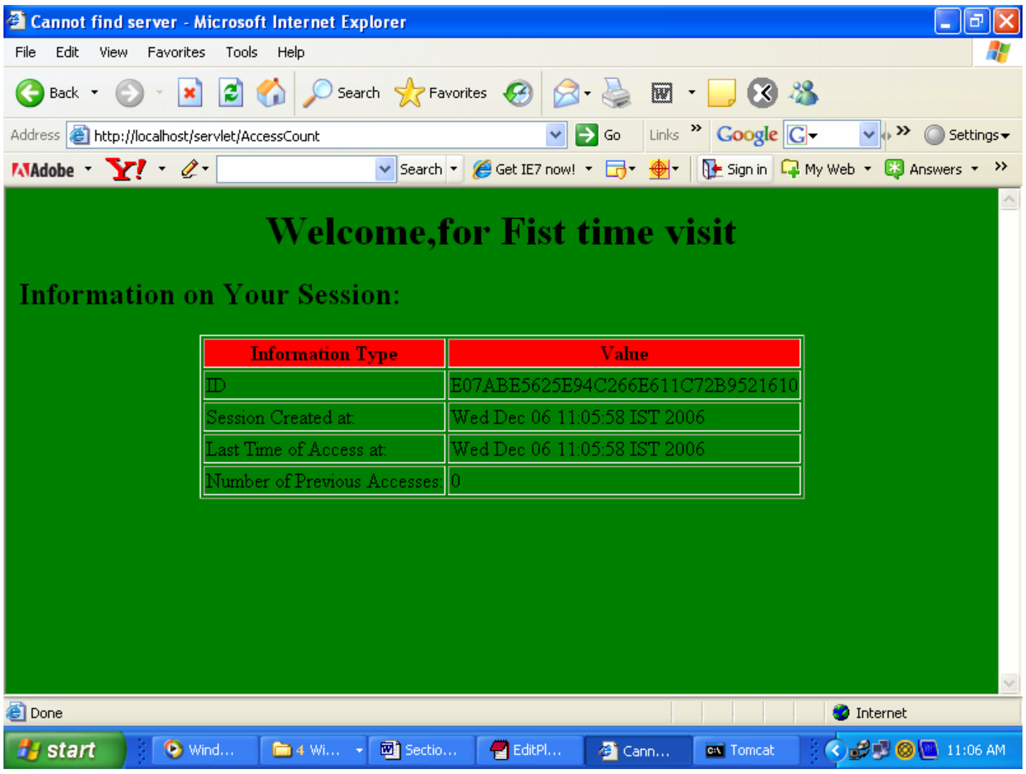
Figure 19a: Output of TestServlet Program

If, you are developing servlet applications which uses Packages and Utilities then, you have to be more careful in terms of putting class files in appropriate directories both during development and when deployed to the server. For example, if, you have a package named myServlet then, compiling your application.*java*, put application.cl*ass* in *install_dir/webapps/ROOT/WEB-INF/classes/myServlet*. Once you have placed the .class file of servlet in the proper directory, access it with the URL *http://localhost/servlet/myServlet.application*.

Now, let us see one more servlet program. This program is for counting the number of times this application is accessed. Here, we will use session tracing for this counting.

Servlet File –AccessCount.java

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

/** An example of session tracking. */

public class AccessCount extends HttpServlet {
 public void doGet(HttpServletRequest request,HttpServletResponse response)
 throws ServletException, IOException
 {
  response.setContentType("text/html");
  PrintWriter out = response.getWriter();
  String title = "Access Count Example By Session Tracking ";
  String heading ;
  //Creating HTTP Session
  HttpSession session = request.getSession(true);
  Integer Count = (Integer)session.getAttribute("Count");
  if (Count == null) {
  Count = new Integer(0);
  heading = "Welcome,for Fist time visit ";
  }
  else {
   heading = "Welcome Again";
   Count = new Integer(Count.intValue() + 1);
  }
  session.setAttribute("Count", Count);
  out.println("<HTML><HEAD><TITLE> Counting by Session
Tracing</TITLE></HEAD>");
  out.println("<BODY BGCOLOR=GREEN>\n" +
       "<H1 ALIGN=\"CENTER\">" + heading + "</H1>\n" +
       "<H2>Information on Your Session:</H2>\n" +
       "<TABLE BORDER=1 ALIGN=\"CENTER\">\n" +
       "<TR BGCOLOR=RED>\n" +
       "<TH>Information Type<TH>Value\n" +
       "<TR>\n" +
       " <TD>ID\n" +
       " <TD>" + session.getId() + "\n" +
       "<TR>\n" +
       " <TD>Session Created at:\n" +
       " <TD>" +
       new Date(session.getCreationTime()) + "\n" +
       "<TR>\n" +
       " <TD>Last Time of  Access at: \n" +
       " <TD>" +
       new Date(session.getLastAccessedTime()) + "\n" +
       "<TR>\n" +
       " <TD>Number of Previous Accesses:\n" +
       " <TD>" + Count + "\n" +
       "</TABLE>\n" +
       "</BODY></HTML>");
 }
}
```
Compile AccessCount.java file and save AccessCoun.class file in C:\Tomcat5.5\apache-tomcat-5.5.15\webapps\ROOT\WEB-INF\classes directory. Now aaccess URL http://localhost/servlet/AccessCount. For the first access of AccessCount page you will get the screen given in *Figure 19b*.


Cannot find server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/servlet/AccessCount Go Links
Done Internet
start 11:06 AM

Welcome,for Fist time visit

Information on Your Session:

Information Type	Value
ID	E07ABE5625E94C266E611C72B9521610
Session Created at:	Wed Dec 06 11:05:58 IST 2006
Last Time of Access at:	Wed Dec 06 11:05:58 IST 2006
Number of Previous Accesses:	0

Figure 19b: Output of AccessCount Program

1.5 XML PROGRAMMING

For writing XML document you can use any text editor. XML files are saved with extension .xml.

As you know XML is a very intuitive general specification for how to declare something to make it hierarchical data, or markup language. XML is a universal standard specified by the W3C. In XML, within its specification, you can specify your own set of rules for naming XML elements, and other elements.

A major advantage of XML is that it can represent anything the human mind can imagine, that to, in the form of which can be easily read /understood by both a human being and a computer (machine).

XML Editors

To create XML documents, you will need a text editor of some kind, such as vi, Macintosh's BBEdit, SimpleText,Windows Notepad,Windows WordPad or any other text editor. XML documents are supposed to be written in Unicode, but in practice, you can write them in ASCII also. While writing XML document just make sure that it is being saved in your editor's plain text format.

However, it can be a lot easier to use an actual XML editor, which is designed specially for the job of handling XML. Some of the XML editors you will find at the following URLs:
Here's a list of some programs that let you edit your XML:

Adobe FrameMaker, http://www.adobe.com

XML Pro, http://www.vervet.com

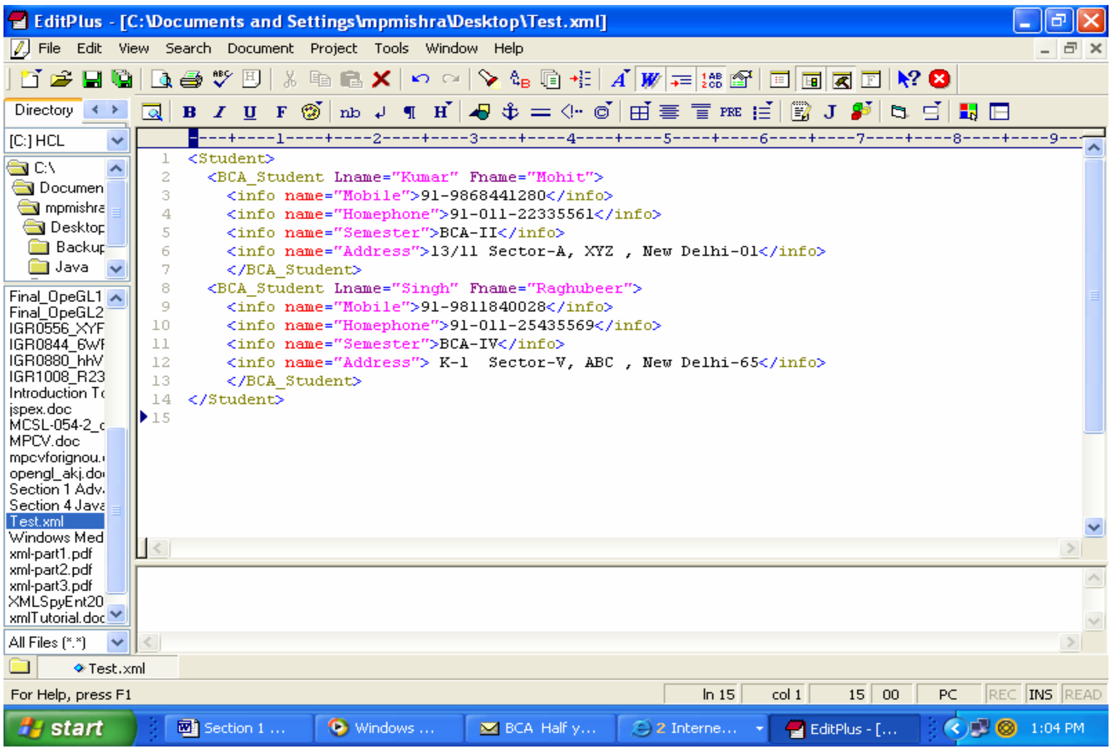
XML Writer, http://xmlwriter.net

XML Notepad, http://msdn.microsoft.com/xml/notepad/intro.asp

XML Spy, http://www.xmlspy.com

You have used EditPlus for Java programming; it can also be used for developing XML Documents. Now, let us see one example of an XML document. To develop this document EditPlus is used.

Here, we are taking the example of a document, which presents personal information of BCA students. This document is saved in Test.xml file. You have to keep in mind that neither Student, BCA_Student, Mobile, Homephone, Semester, Address, info nor name are keywords:


EditPlus - [C:\Documents and Settings\mpmishra\Desktop\Test.xml]
File Edit View Search Document Project Tools Window Help
Directory
[C:] HCL
Test.xml
For Help, press F1
ln 15 col 1 15 00 PC REC INS READ
start
1:04 PM

```
<Student>
  <BCA_Student Lname="Kumar"  Fname="Mohit">
    <info name="Mobile">91-9868441280</info>
    <info name="Homephone">91-011-22335561</info>
    <info name="Semester">BCA-II</info>
    <info name="Address">13/11 Sector-A, XYZ , New Delhi-01</info>
    </BCA_Student>
  <BCA_Student Lname="Singh"  Fname="Raghubeer">
    <info name="Mobile">91-9811840028</info>
    <info name="Homephone">91-011-25435569</info>
    <info name="Semester">BCA-IV</info>
    <info name="Address"> K-1  Sector-V, ABC , New Delhi-65</info>
    </BCA_Student>
</Student>
```

Figure 20: An XML Document in EditPlus

Some Concepts

While writing XML documents take care of the following. Though it is discussed in the course MCS-051, it will be very useful while writing XML documents:

i) Similar to HTML, start tags are angle bracket enclosed, and end tags are angle bracket enclosed with a prepended forward slash.

ii) An entity started by a start tag and ended by an end tag is called an *element*.

iii) Elements can contain other elements. In our example, the Student element contains two BCA_Student elements.

iv) An element can contain a mix of different elements.

v) An XML file must contain exactly one element at the top level. In our example, the Student is the top level.

As you know XML, allows the programmer/author to define their own tags and own document structure.
While defining your own tag you should take care of that:

- XML Tags are Case Sensitive:

For example With XML, the tag <myLetter> is different from the tag <myletter>.

- Opening and closing tags must therefore, be written with the same case:
For example, i is incorrect and ii is correct.

  i. <Message>This is the incorrect version </message>
  ii. <Message>This is the correct version </message>
- XML Attribute Values Must be Quoted:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note date=12/11/2006>
<to>Ravi</to>
<from>Raju</from>
</note>
```

is incorrect because the date value is not in double quote (“ ”), to make it correct you have to write the following :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note date="12/11/2006">
<to>Ravi</to>
<from>Raju</from>
</note>
```

XML Browsers

Any XML browser should support a style language such as CSS or XSL. It should also support a scripting language such as JavaScript. Some browsers that support XML are :

Internet Explorer 6

The Internet Explorer is currently one of the most powerful XML browsers available. The Internet Explorer 6.0 can display XML documents directly. It can also handle XML in scripting languages (JScript, Microsoft's version of JavaScript, and Microsoft's VBScript are supported). The Internet Explorer 6.0 and Windows XP is based on both the W3C XSLT 1.0 and the W3C XPath 1.0 Recommendations.

Netscape Navigator 6

Netscape Navigator 6 or later has some XML support. Netscape Navigator support for stylesheets. Netscape Navigator 6 also supports the XML-based User Interface Language (XUL), which lets you configure the browser.

Jumbo

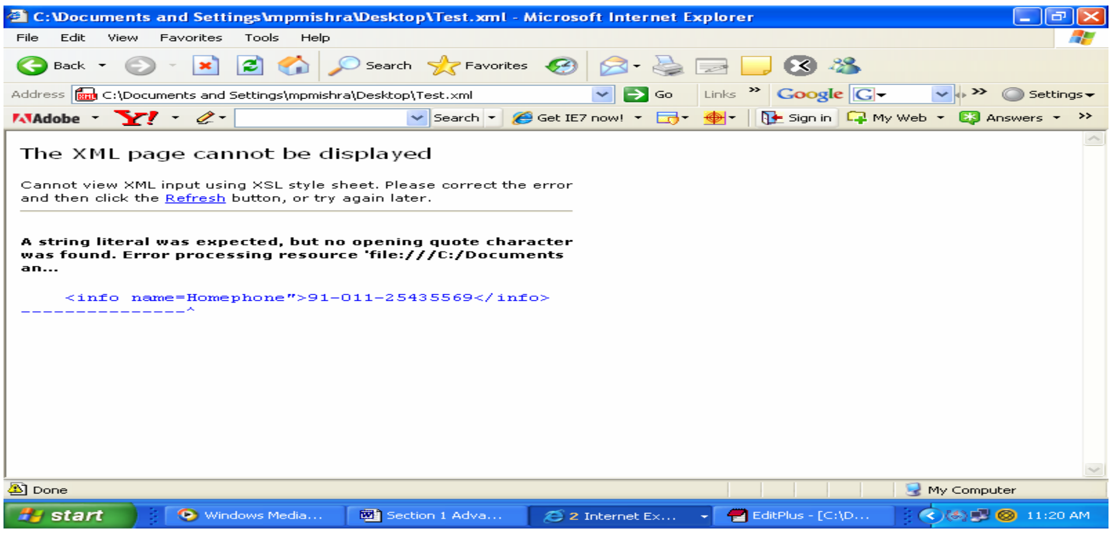
Jumbo, is one of the famous XML browser. It is designed to work with XML and other Chemical Markup Language (CML). You can find Jumbo for free at http://www.xml-cml.org/. Internet Explorer 6

Viewing XML Files In Internet Explorer:

You can open the XML file (typically by clicking on its link). The XML document will be displayed with colour-coded root and child elements. A plus (+) or minus sign (–) to the left of the elements can be clicked to expand or collapse the element structure.

If you want to view the XML source (without the + and - signs), then go to select "View Page Source" or "View Source" from the browser menu.

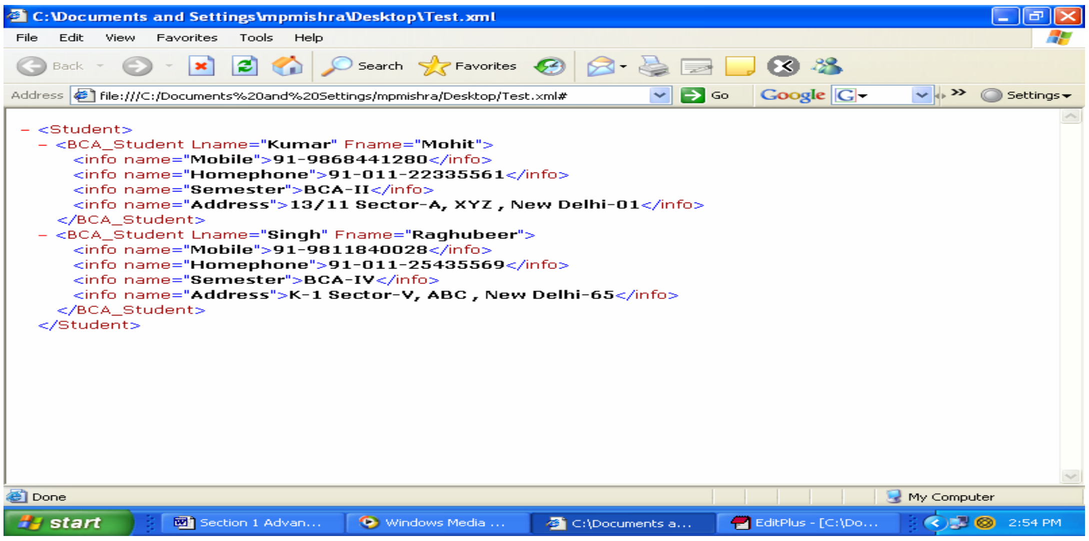
Note: If you try to open an erroneous XML file (XML file with some error), the browser will report the error, similar to that shown in *Figure 21*.


C:\Documents and Settings\mpmishra\Desktop\Test.xml - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\mpmishra\Desktop\Test.xml
Go Links Google Settings
Adobe Search Get IE7 now! Sign in My Web Answers
Done My Computer
start Windows Media... Section 1 Adva... 2 Internet Ex... EditPlus - [C:\D... 11:20 AM

```
The XML page cannot be displayed

Cannot view XML input using XSL style sheet. Please correct the error
and then click the Refresh button, or try again later.

A string literal was expected, but no opening quote character
was found. Error processing resource 'file:///C:/Documents
an...

     <info name=Homephone">91-011-25435569</info>
----------------^
```

Figure 21: Viewing XML Document with Internet Explorer Error

If you see Test.xml file using Internet Explorer you will find following screen:


C:\Documents and Settings\mpmishra\Desktop\Test.xml
File Edit View Favorites Tools Help
Back Search Favorites
Address file:///C:/Documents%20and%20Settings/mpmishra/Desktop/Test.xml#
Go Google Settings
Done My Computer
start Section 1 Advan... Windows Media ... C:\Documents a... EditPlus - [C:\Do... 2:54 PM

```
- <Student>
  - <BCA_Student Lname="Kumar" Fname="Mohit">
      <info name="Mobile">91-9868441280</info>
      <info name="Homephone">91-011-22335561</info>
      <info name="Semester">BCA-II</info>
      <info name="Address">13/11 Sector-A, XYZ , New Delhi-01</info>
    </BCA_Student>
  - <BCA_Student Lname="Singh" Fname="Raghubeer">
      <info name="Mobile">91-9811840028</info>
      <info name="Homephone">91-011-25435569</info>
      <info name="Semester">BCA-IV</info>
      <info name="Address">K-1 Sector-V, ABC , New Delhi-65</info>
    </BCA_Student>
  </Student>
```

Figure 22: Viewing XML Document in Internet Explorer

XML Parsers

You need parsers to break up your document into its component pieces and make them accessible to other parts of a program. Some parsers check for well-formedness, and some check for document validity. However, if you just want to check that your XML is both well formed and valid, all you need is an XML *validator*.

Microsoft's XML Parser

Microsoft has an XML parser which is a COM component. This parser, comes with Internet Explorer 5 and higher versions. Once you have installed the Internet Explorer, the parser is available to scripts.This XML parser supports all the necessary functions to traverse the node tree, access the nodes and their attribute values, insert and delete nodes, and convert the node tree back to XML. MSXML Parser 2.5 is the XML parser that come with Windows 2000 and IE 5.5 and MSXML Parser 3.0 is the XML parser that is shipped with IE 6.0 and Windows XP.

Here is a list of some of the other XML parsers:

i) SAX: The Simple API for XML—SAX is a well-known parser written by David Megginson, et al. (http://www.megginson.com/SAX/index.html)

ii) expat—This is a famous XML parser written in the C programming language by James Clark (http://www.jclark.com/ xml/expat.html).

iii) LT XML—This is an XML developers toolkit developed by the Language Technology Group at the University of Edinburgh (http://www.ltg.ed.ac.uk/software/xml/).

iv) XML for Java (XML4J)—From IBM AlphaWorks), this is one of the very widely used XML parser that adheres well to the W3C standards. (http://www.alphaworks.ibm.com/tech/xml4j)

v) XP—This is a nonvalidating XML processor, written in Java by James Clark (http://www.jclark.com/xml/xp/index.html).

vi) SXP Silfide XML Parser (SXP)—This is an XML parser. Basically it is a complete XML application programming interface (API) for Java (http://www.loria.fr/projets/XSilfide/EN/sxp/).

1.6 LIST OF LAB ASSIGNMENTS

Session 1:

Exercise 1. Verify installation and setting of Web container/Web Server/Tomcat and prepare an installation report, which contains setting of class path, server port, starting and shutting down of server etc.

Exercise 2. Write a servlet program which overrides doPost( ) method and write “ Welcome to Servlet Programming”. Also create an HTML table with five rows and two columns in this program.

Session 2:

Exercise 3. Write a servlet program that takes your name and address from an HTML Form and displays it.

Exercise 4. Write a servlet program that displays current date and time. Your servlet should also indicate the number of times it has been assessed since it has been loaded.

Exercise 5. Write a program to show inter servlet communication between two servlets.

Session 3:

Exercise 6. Write a Servlet Program that displays server information (server name, port etc.).

Exercise 7. Write a program, using servlet and JDBC which takes students roll number and provides student information, which includes the name of the student, the address, email-id, program of study, and year of admission. You have to use a database to store student's information.

Exercise 8. Write program of **Exercise 7** with login and password protection. Display a message if login and password are not correctly given.

Session 4:

Exercise 9. Write a program using servlet and JDBC for developing an online application for the shopping of computer science books. (Hint: use concept of session tracking) You have to create a database for book title, author(s) of book, publisher, year of publication, price. Make necessary assumptions for book shopping.

Exercise 10. Write a JSP program to output, "Welcome to JSP world. The time now is: system *current time*. Use a scriptlet for the complete string, including the HTML tags.

Exercise 11. write a JSP page that display a randomly generated number in first visit to this page and repeat displaying this same number in subsequent visits.

Session 5:

Exercise 12. Write a JSP page to output the values returned by System.getProperty for various system properties such as java.version, java.home, os.name, user.name, user.home, user.dir etc. Each property should be displayed in a separate row.

Exercise 13. Write a JSP page to use either <jsp:include> or <jsp:forward> depending upon the value of a Boolean variable.

Exercise 14. Write a JSP page using <jsp:forward> to go to a servlet program which display your name, date of birth and address.

Session 6:

Exercise 15. Create an HTML form to take customer information (Name, Address, Mobile No.). Write a JSP program to validate this information of customers.

Exercise 16. Write a JSP program using <jsp:include> to include the program written in Exercise 9.

Exercise 17. Create a database of students who are in the 5th Semester of the MCA program with you at your study center. Write a program using JSP and JDBC to display the name and address of those students who are born after 1985.

Session 7:

Exercise 18. Write a JSP program which display a web page containing your personal information such as: Name, Date of Birth, Sex, Area of Interest, Specialisation and a paragraph explaining what you want to be in the next five years.

Exercise 19. Develop an application that collects/maintains the product information of an electronics goods production company, in a database. Write a JSP page to retrieve (to display) selective this information in database on demand. Make necessary assumptions to develop this application.

Session 8:

Exercise 20.The following example is a note to Ramu from Deenu, stored in XML:

```
<note>
<to> Ramu </to>
<from>Deenu</from>
<heading>Festival  Wishes</heading>
<body>May God give you all the happiness</body>
</note>
```

Extend this document by adding some more information in this document.

Exercise 21. Imagine that this is the description of a book:

My First Servlet

Introduction to Servlet

- What is Servlet
- What are the advantages of Servlet

Servlet Life Cycle

- Servlet Initialisation
- Servlet Realoading
- Destroying a Servlet

Write an XML document that describes this book.

Exercise 22. Write an XML program to show that white space is preserved in XML.

Session 9:

Exercise 23. Imagine that, you are looking at an HTML document with the following table for students and their subjects:

Visual Basic Programming	B.Tech	Mohan	Naveen
Java Programming	M.Tech	Robert John	Sudhansh
ASP Programming	MSc	Neeta	Ravi

Write an XML document to represent this information.

Exercise 24.Use the plus sign and arrows to open and close the different elements of the following XML files:

i) http://www.w3schools.com/xml/cd_catalog.xml
ii) http://www.w3schools.com/xml/plant_catalog.xml

Exercise 25. Make an HTML form that name of a table, number of rows and number of columns in this table, using JSP page display the table of given specification.
Session 10:

Exercise 26. Using Servlet, JSP, JDBC, and XML, create a Web application for a recruitment agency to providing assistance in searching the candidates from its databases as per requirements of its various clients. Make necessary assumptions while developing this application.

1.7 FURTHER READINGS

1. Head First Servlet and JSP By Bryan Basham, Kathy Sierra, and Bert Bates, *O. Reilly publication First Edition Eighth India Reprint- 2006.*

2. http://www.w3schools.com/xml/default.asp

3. http://www.w3.org/XML/

4. http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/Servlets.html

5. http://tomcat.apache.org/

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