WebSphere. software

IBM

Redbooks Paper

Javier Voos

WebSphere Application Server V6: Web Container Problem Determination

If users receive unexpected results in a Web browser (such as errors or incorrect information), you might have a problem with the Web components in the application. The runtime environment for Web components is called the *Web container*. This paper discusses techniques for diagnosing problems that can occur in the Web container. Potential symptoms of a Web container problem can include:

- Users are not able to access a Web component
- Unexpected behavior when running a Web component
- Errors when starting a Web component
- Problems with JSP™ compilation
- Errors or exceptions thrown when running a Web component
- Messages starting with SRVE (Web container), JSPG (JSPs), or JSFG (JSF)

Problems areas that are related to the Web container also include session management problems and character encoding problems.

Important: We recommend that you start your problem determination process by reading *Approach to Problem Determination in WebSphere Application Server V6* at http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf.

Introduction

The Web container is a runtime environment for Web applications. It processes servlets, JSP files, and other types of server-side includes. The Web container also provides session management, static content processing and an inbound transport chain for HTTP requests. Each application server runtime has one logical Web container, which can be modified but not created or removed.

Figure 1 illustrates the Web container and its place within the application server.

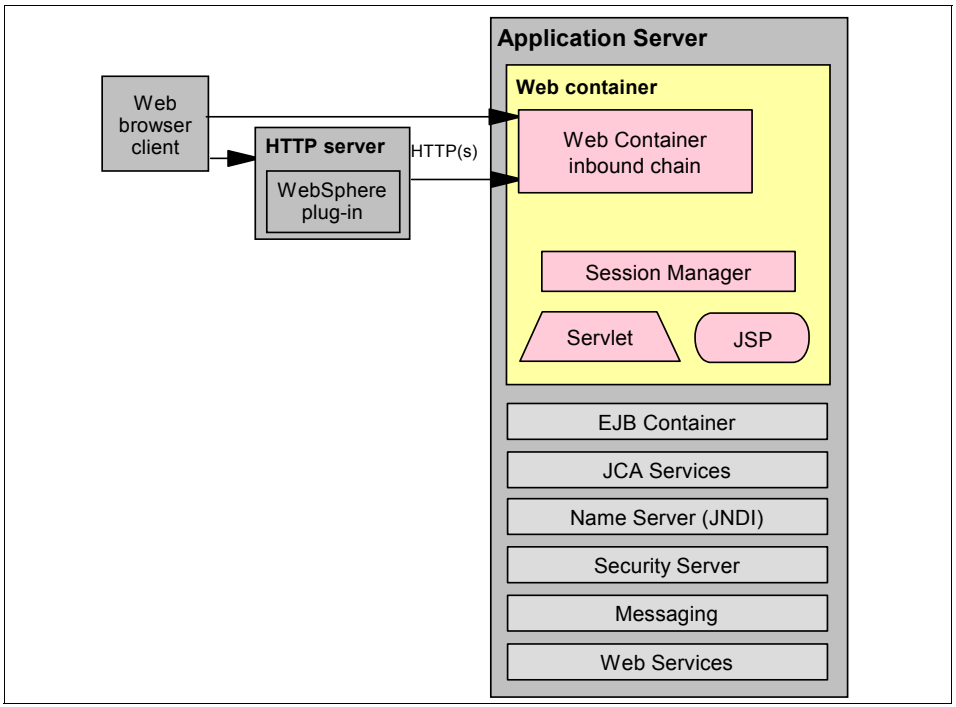

Figure 1 Web container overview

Each Web container provides the following:

- Web container transport chains

  Requests are directed to the Web container using the Web container inbound transport chain. The chain consists of a TCP inbound channel that provides the connection to the network, an HTTP inbound channel that serves HTTP 1.0 and 1.1 requests, and a Web container channel over which requests for servlets and JSPs are sent to the Web container for processing.

- Servlet processing

  When handling servlets, the Web container creates a request object and a response object and then invokes the servlet service method. The Web container invokes the servlet's destroy method when appropriate and unloads the servlet, after which the JVM™ performs garbage collection.

- HTML and other static content processing

  Requests for HTML and other static content that are directed to the Web container are served by the Web container inbound chain. However, in most cases, using an external Web server and Web server plug-in as a front-end to a Web container is more appropriate for a production environment.

- Session management

  Support is provided for the javax.servlet.http.HttpSession interface as described in the Servlet API specification.

Web applications

Servlets and JavaServer™ Pages™ (JSP) files are referred to as Web components. Static content files (such as HTML pages, image files, and XML files) are bundled with Web components during application assembly to create a Web module. A Web module is the single deployable and usable unit of Web resources and has a specific structure or archived format known as a *Web archiv*e (WAR) file. A J2EE™ Web module corresponds to a Web application as defined in the Java™ Servlet specification.

The top-level directory of a Web module is the *document root* of the application. The document root is where JSP pages and static Web resources are stored. The document root contains a subdirectory named /WEB-INF/, which contains the Web application deployment descriptor (web.xml) and the server-side classes used by the module.

Figure 2 on page 4 illustrates the directory structure of a Web application.

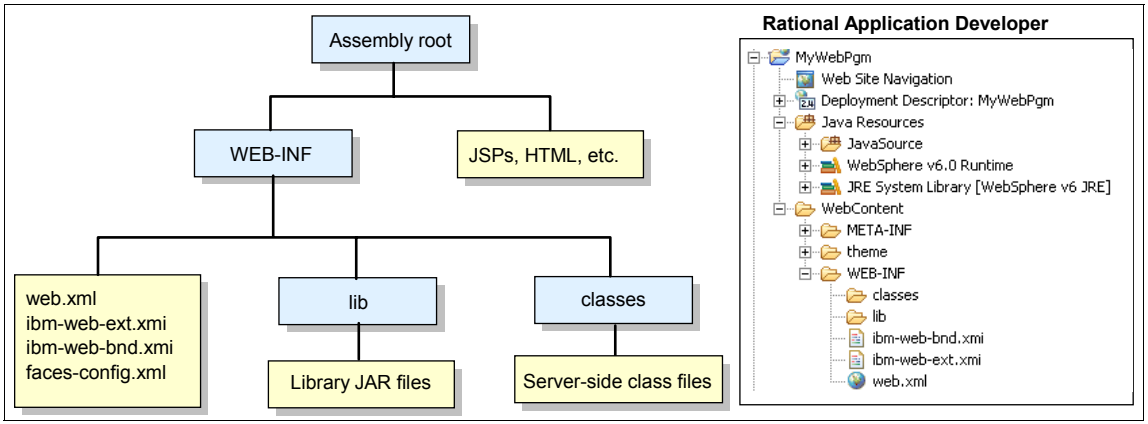

Figure 2 Web application components

Initial symptoms

A problem that occurs in the Web container usually causes unexpected results to be displayed in the Web browser. Three common indications are addressed here:

- HTTP 404 errors
- HTTP 500 errors
- Incorrect information

Note: If the application has an error page configured in the Web module to be delivered when HTTP errors occur, the user will see this page instead of an HTTP error page. In this case, the true error might not be apparent until the Web server log files are examined.

Figure 3 on page 5 shows a flow chart of the high-level symptoms and the potential problem areas that might apply to each.

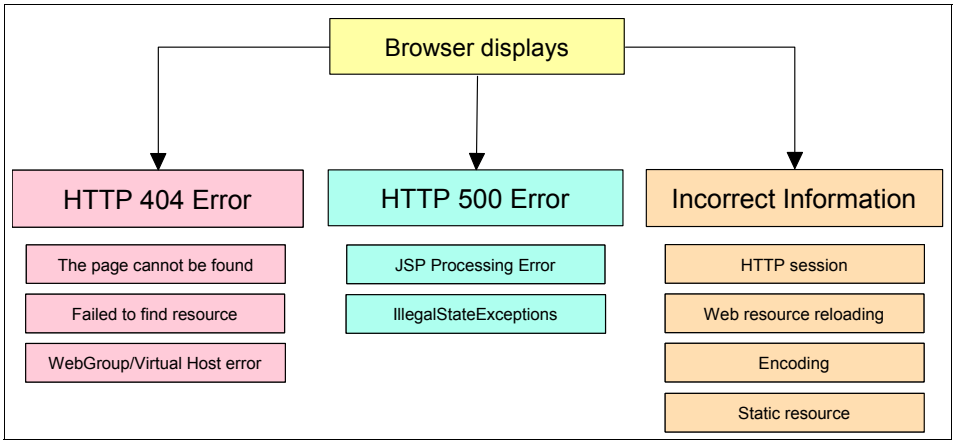

Figure 3 Initial symptoms for Web container problems

Work the problem

You begin the problem determination process by collecting the appropriate data that is required to diagnose the problem. We give you a list of all the documentation that might be required and how to collect it. If you have limited ability to recreate the problem, you might want to collect all the documentation at once, before starting the problem determination process.

Next, you go through a series of questions and actions that will help you define the high-level symptoms that you are experiencing. Each of these steps leads you to a more detailed procedure that takes you through the process of collecting and analyzing data to determine the most likely source of the problem.

And lastly, we provide guidance on the next step to take for resolution, whether it be a support site, contacting IBM®, information about configuration, or some other suggestion as to how to proceed.

Data to collect

The following data will help you with the problem determination process in a Web container execution environment:

- WebSphere® Application Server JVM logs: SystemOut and SystemErr files
- WebSphere Application Server process logs: native_stderr.log and native_stdout.log log files
- Web server log files

For information about collecting the JVM and Process logs, see *WebSphere Application Server V6: Diagnostic Data* at:

http://www.redbooks.ibm.com/redpapers/pdfs/redp4085.pdf

The Web server log files names and locations are specific to the product (IBM HTTP Server, Apache, SunOne, IIS, and Domino®) that is used for this function.

If the problem is difficult to recreate or disruptive to business operations, see “The next step” on page 45 for a complete list of documentation to collect before continuing. In particular, you should review the MustGather documents for a complete list of documentation that is required by IBM support.

High-level symptom analysis

Consider the following error types. If you do not find the description of your problem here, go to “The next step” on page 45.

What to look for

Many indicators of a Web container problem first appear to a user as an unexpected Web browser page or page contents. The unexpected information can be an error page that results from an HTTP error or a page that has incorrect or incomplete information displayed.

When an HTTP error occurs, the Web browser displays the error page with the error code (in particular, HTTP error codes 404 and 500 are likely). If a custom error page has been configured for the Web module, you need check the Web server logs files to determine the specific HTTP error code.

HTTP 404 errors

HTTP 404 errors can have different underlying causes. Table 1 shows the HTTP 404 errors that we address in this paper.

Table 1 Where to go for information about HTTP 404 errors

If the errors include	Then go to
`Page can't be found or displayed`	“Symptom: HTTP 404 error - The page cannot be displayed” on page 8
`JSPG0036E: Failed to find resource message`	“Symptom: HTTP 404 error - Failed to find resource” on page 11
`WebGroup/virtual host not define`	“Symptom: HTTP 404 error - WebGroup/virtual host not defined” on page 13

HTTP 500 errors

HTTP 500 errors can also have different underlying causes, which include the following:

- If the errors indicate JSP processor errors or incorrect syntax in JSPs, go to “Symptom: HTTP 500 error - JSP processing error” on page 15.
- If the symptoms include an HTTP 500 error page on the browser and in the SystemOut.log file you see an `IllegalStateException`, go to “Symptom: HTTP 500 error - IllegalStateException” on page 17.

Incorrect information displayed on Web pages

If the users are receiving the correct page on the Web browser but the page contains invalid information, check these symptoms:

- Users report that pages appear with missing elements.

  If a Web page appears but is missing static resources such as text, images, or file segments, go to “Static resources not displayed” on page 26.

- Users report seeing an old version of application pages.

  If a JSP file has been modified and deployed to the server but the changes do not appear to the browser interface, you might need to ensure that the application has been enabled for JSP reloading. Go to “Web resources not reloading” on page 28.

- Users report that pages contain invalid information such as multiple question marks (???) or garbage characters.

  It is possible that the application uses DBCS characters (Japanese, Chinese, Korean languages) or specific characters for other languages that are not included in the default character encoding.

  In these cases, a correct character encoding configuration is necessary to display and process this information without problems. For more details related to encoding configurations, go to “Encoding and internationalization issues” on page 31.

- Users report problems such as lost data during a session.

  Users repeatedly have to enter information that should be saved during the session, lose shopping cart information, or have other short-term data loss.

  This data loss might be caused by a problem with HTTP sessions. In a clustered environment, the problem could be with the Web server plug-in. However, there are other possible causes, such as configuration or application errors.

  To further analyze this problem, go to “HTTP session management” on page 38.

Symptom: HTTP 404 error - The page cannot be displayed

Figure 4 shows an overview of the steps that you can take to find the cause of an HTTP 404 error caused by a page cannot be displayed condition.

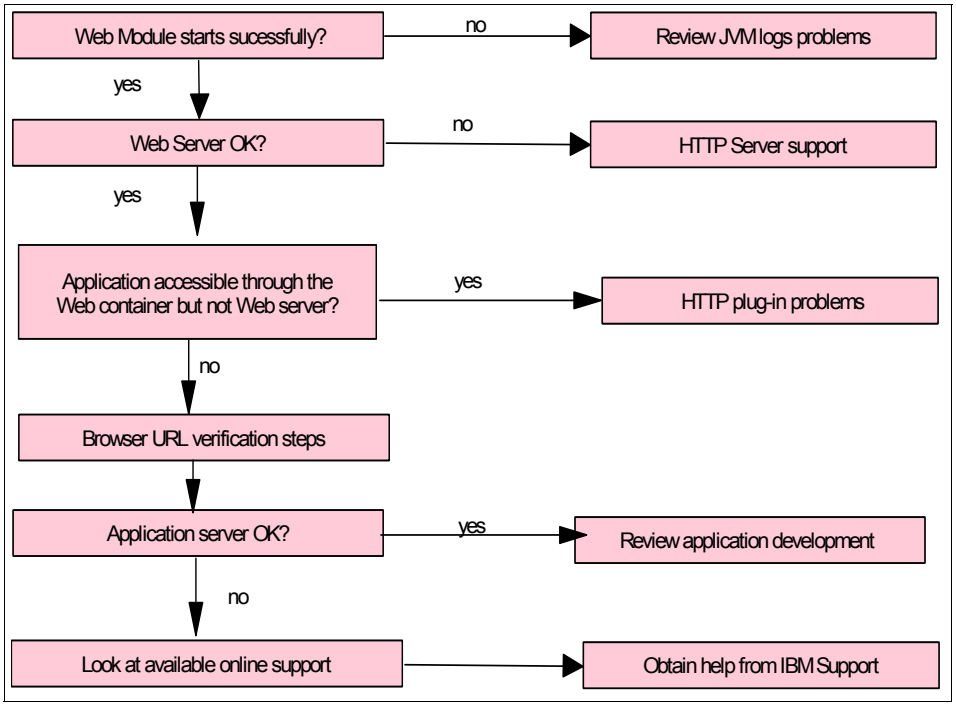

Figure 4 Page cannot be displayed verification steps

Verify that the Web module has started successfully

Check the JVM logs for the application server, and look for messages in the SystemOut.log file (Example 1) to verify that the Web container and Web module started successfully.

Example 1 Web Module start process results in SystemOut.log file

```
ApplicationMg A   WSVR0200I: Starting application: [application_name]
WebContainer  A   SRVE0161I: IBM WebSphere Application Server - Web Container.
Copyright IBM Corp. 1998-2004
WebContainer  A   SRVE0162I: Servlet Specification Level: 2.4
WebContainer  A   SRVE0163I: Supported JSP Specification Level: 2.0
WebGroup      A   SRVE0169I: Loading Web Module: [web_module_name]
ApplicationMg A   WSVR0221I: Application started: [application_name]
```

One possible reason that would keep the Web container from starting is a problem with the session manager. To determine if this is the case:

- Look for any errors or exceptions containing a package name of com.ibm.ws.webcontainer.httpsession. You will normally find these errors between the `starting application` message and the `application started` message.
- Look in the logs for session manager related messages. These messages will be in the format `SESNxxxxE` for errors and `SESNxxxxW` for warnings. You might also see transaction messages that indicate a problem with session data (see Example 2).

Example 2 Session manager message error

```
Error "SRVE0054E: An error occurred while loading session context and Web
application.
```

You can look up the extended error definitions in the session manager message table:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.messages.doc/doc/SESN.html`

If you find a session manager error but the explanation is not sufficient to solve the problem, go to “The next step” on page 45.

Verify that the Web server is working

Verify that the Web server is healthy by accessing the URL http://*server_name* from a browser and seeing whether the Welcome page appears. This action indicates whether the Web server is up and running, regardless of the state of WebSphere Application Server.

If the Web server Welcome page does not appear (that is, if you get a browser message such as page cannot be displayed or something similar), the problem is most likely in the Web server. For information about how to approach Web server problems, see *Approach to Problem Determination in WebSphere Application Server V6* at:

`http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf`

Verify that the Web server plug-in is working

If you have recently updated or installed the application, ensure that the Web server plug-in was regenerated and propagated to the Web server. Also, ensure that the Web server is using the new plug-in configuration file.

If you have regenerated the plug-in and are sure it is in use but still have a problem, you can bypass the Web server and access the application directly

from the application server. This is not the recommended method of serving a production Web site. However, it provides a good diagnostic tool when it is not clear whether a problem resides in the Web server, WebSphere Application Server, or the Web server plug-in.

Note: To access the application directly through the Web container, do the following:

1. Find the port for the Web container:
   a. In the WebSphere administrative console, select **Servers** → **Application Servers**.
   b. Select the server name.
   c. Under the Communications section, expand Ports.
   d. Use the port number listed for WC_defaulthost.
2. Use the port number to access the resource from a browser. For example, if the port is 9080, the URL is:

   ```
   http://hostname:9080/myAppContext/myJSP.jsp
   ```

If you can access the application via the application server but not through the Web server, you are most likely experiencing a problem with the Web server plug-in. In this case, consult *WebSphere Application Server V6: Web Server Plug-in Problem Determination* at:

http://www.redbooks.ibm.com/redpapers/pdfs/redp4045.pdf

Verify that the URL is correct

If you cannot access the page directly from the application server, verify that the URL being used to access the page is correct. For more information, see “Application URL specification” on page 22.

Verify that the application server is working

If the URL appears to be correct but you cannot access the resource directly through the application server, verify the health of the hosting application server and Web module by doing the following:

1. View the hosting application server and Web module in the administrative console to verify that they are up and running. In a single server environment, you can check the application server process to see if it is running, or you can use the **serverStatus** command as follows:

   ```
   c:\cd WebSphere\AppServer\profiles\AppSrv01\bin
   serverStatus server1
   ```

2. Copy a simple HTML or JSP file to the Web module document root and try to access it. The Web module document root is located in:

   <WAS_install_root>/profiles/*<profile>*/installedApps/*<node>*/*<application>*.ear*<web module>*.war

   If successful, the problem is with the resource. View the SystemOut.log file for the application server to discover why the resource cannot be found or served.

Look at available online support

If none of these steps fixes your problem, check to see if the problem has been identified and documented by looking at the available online support (hints and tips, tech notes, and fixes) that are related to Web container problems:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDF

If the searches fail to find the problem, then go to “The next step” on page 45 for information about gathering the MustGather documentation for HTTP status code 404 “Not Found” problems.

Symptom: HTTP 404 error - Failed to find resource

If the browser displays a `JSPG0036E: Failed to find resource` message, the JSP processor cannot find the specified JSP page in the Web module, as shown in Figure 5 on page 12.

Figure 5 JSPG0036E: Failed to find resource

Verify that the URL is correct

Follow the steps described in “Application URL specification” on page 22 to see if the URL being specified is the correct URL.

If the URL is incorrect and is being created as a link from another JSP file, servlet, or HTML file, try correcting it in the browser URL field and reloading the page. This ensures that you know the correct URL before updating the application to use it.

If the URL appears to be correct, it is possible that this page does not exist in the Web module deployment unit (WAR file). Verify that the requested page is located in the Web module directory structure:

<WAS_install_root>/profiles/*<profile>*/installedApps/*<node>*/
<application>.ear*<web module>*.war

Look at available online support

If these steps have failed to identify the problem, see “The next step” on page 45.

Symptom: HTTP 404 error - WebGroup/virtual host not defined

The error message `SRVE0017W: WebGroup/Virtual Host has not been defined` can appear in the SystemOut.log file for many reasons. Check the following to correct the problem.

Verify that the Web module started

Verify that the Web module has started successfully (see “Verify that the Web module has started successfully” on page 8).

Verify that the host alias is unique

A duplicate host alias that is defined in multiple virtual hosts can cause this type of problem. For example, in Example 3, the host alias `test:80` is duplicated in both virtual hosts because a URI that contains `test:80` would match aliases in both virtual hosts (`*:80` and `test:80`).

Example 3 Virtual host definitions

```
<VirtualHostGroup Name="default_host">
       <VirtualHost Name="*:80"/>
       <VirtualHost Name="*:9080"/>
</VirtualHostGroup>

<VirtualHostGroup Name="test_host">
       <VirtualHost Name="test:80"/>
       <VirtualHost Name="*:9081"/>
</VirtualHostGroup>
```

To resolve this problem:

1. In the WebSphere administrative console, click **Environment** → **Virtual Hosts**.
2. Select the target virtual host, then click **Host Aliases**, and select the *:80 host alias.
3. Change the Host Name field to a specific alias instead of * alias.
4. Restart the application server.

Note: The existence of localhost as an alias can cause duplicate entries.

If localhost is used as an alias entry, check the etc/hosts file to ensure that all host names (aliases) associated with the loop back address (127.0.0.1) are part of the same virtual host grouping (for example, default_host).

Verify that the URL is correct

See “Application URL specification” on page 22 to determine that the URL is correct.

Note: The virtual host matching that is performed by the application server is case sensitive. For example, if your virtual host alias is listed as myserver:80, a request to http://MYSERVER:80/snoop results in the following message in the browser:

```
SRVE0017W: A WebGroup/Virtual Host to handle MYSERVER:80 has not been
defined.
```

Favicon.ico reference

When accessing the WebSphere administrative console, you could see an error similar to the following:

```
Servlet Request Processor Exception: Virtual Host/WebGroup Not Found : The web
group /favicon.ico has not been defined.
```

If you receive this error message, refer to Technote # 1193379 at the following Web address for the cause and solution details:

http://www.ibm.com/support/docview.wss?uid=swg21193379

Look at available online support

If none of these steps fixes your problem, check to see if the problem has been identified and documented by looking at the available online support (hints and tips, technotes, and fixes) that are related to Web container problems:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDF

If your searches fail to find the problem, then go to “The next step” on page 45 for information about gathering the MustGather documentation for servlet engine and Web container problems.

Symptom: HTTP 500 error - JSP processing error

Figure 6 shows the diagnostic steps to take when the symptoms of the problem include an HTTP 500 - JSP processing error.

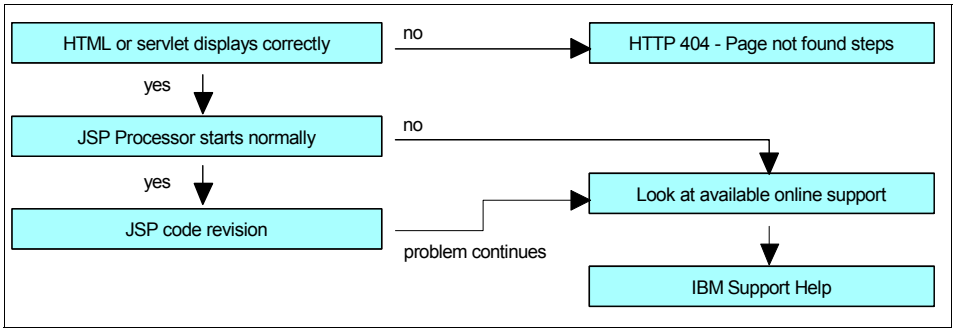

Figure 6 JSP processing error

Determine if the failing file is the only failure

You should determine whether other resource types such as HTML files or servlets are being requested and displayed correctly. If the failure does not appear to be limited to one JSP file, go to “Symptom: HTTP 404 error - The page cannot be displayed” on page 8.

Ensure that the JSP processor started normally

If other resources are being displayed correctly, determine whether the JSP processor has started normally. Browse the SystemOut.log file of the server that hosts the JSP files that you are trying to access. The messages shown in Example 4 indicate that the JSP processor has started normally.

Example 4 JSP processor messages in SystemOut.log file

```
WebContainer  A   SRVE0239I: Extension Factory [class
com.ibm.ws.jsp.webcontainerext.JSPExtensionFactory] was registered
successfully.
WebContainer  A   SRVE0240I: Extension Factory [class
com.ibm.ws.jsp.webcontainerext.JSPExtensionFactory] has been associated with
patterns [*.jsp *.jspx *.jsw *.jsv ].
```

If the JSP processor fails to load, go to “The next step” on page 45 for information about gathering the MustGather documentation for JSP exceptions.

Look for JSP code errors

If the JSP processor starts normally, the problem might be with the JSP file itself. The JSP might have invalid JSP syntax that cannot be processed by the JSP processor. To look for JSP code errors:

- Examine the SystemOut.log file of the target application server for invalid JSP directive syntax messages. Errors similar to that shown in Example 5 indicate this kind of problem.

Example 5 JSP directive syntax error message

```
Message: /test.jsp(2,1)JSPG0076E: Missing required attribute page for jsp
element jsp:include
```

- Examine the SystemOut.log file for problems with invalid Java syntax. Errors that contain text similar to "`failed to compile` (shown in Example 6) indicate this kind of problem.

Example 6 Invalid Java syntax error message

```
com.ibm.ws.jsp.JspCoreException: JSPG0049E: /test.jsp failed to compile :
JSPG0091E: An error occurred at line: 16 in the file: /test.jsp
JSPG0093E: Generated servlet error from file: /test.jsp
```

- Look up the extended error definitions for JSP messages at:

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.messages.doc/doc/JSPG.html

- Correct the error in the JSP file and retry the file.

Look at available online support

If you have not identified the problem, check to see if the problem has been documented by looking at the available online support (hints and tips, technotes, and fixes) for JSP problems at:

http://www-1.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCC2GL&rankprofile=8&dc=DB520+D800+D900+DA900+DA800&dtm

If the searches fail to resolve your problem, go to "The next step" on page 45 for information about gathering the MustGather documentation for JSP exceptions.

Symptom: HTTP 500 error - IllegalStateException

If you have an illegal state exception error, the exception should give you an indication of the illegal state causing the problem. This section addresses the following:

- Invalid session object
- Response generation problems

Invalid session object

The session manager component uses the HttpSession interface to create a session between an HTTP client and the server. When a new session object is created, a unique session ID is assigned to it. The session ID, which is then passed as part of every request, matches the user with the session object. Session tracking gives servlets the ability to maintain state and user information across multiple requests.

Sessions might get invalidated automatically due to a session timeout (see “Session timeout interval” on page 43) or can be ended explicitly by application code. The HttpSession interface provides the following method to terminate a session explicitly:

```
public void invalidate();
```

When a session terminates, the session object and the information that is stored in it are lost permanently. The session manager unbinds any objects bound to the session before it destroys the session.

Figure 7 illustrates the HttpSession interface life cycle.

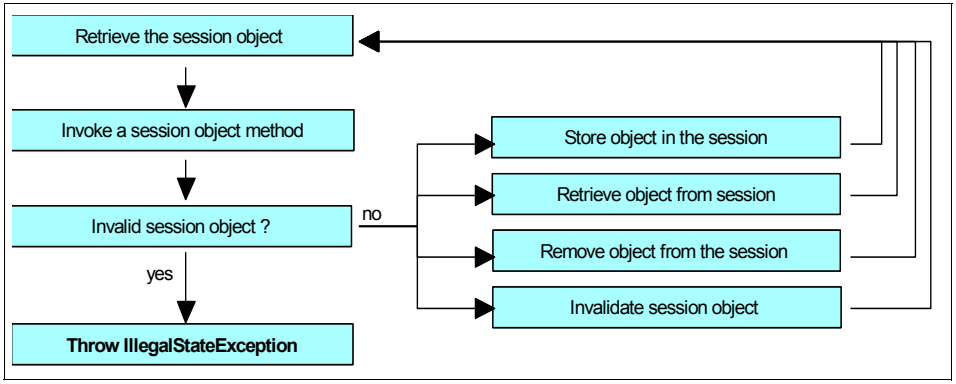

Figure 7 HttpSession interface life cycle

When the application tries to use an invalidated session object, the IllegalStateException occurs, as shown in Example 7.

Example 7 IllegalStateException in invalid session object

```
[7/7/05 16:41:30:627 ART] 00000028 ServletWrappe E   SRVE0068E: Could not
invoke the service() method on servlet TestServlet. Exception thrown :
java.lang.IllegalStateException:
Session Object Internals:
id : pi55X7syi-ExTjyyhFn5Cu7
hashCode : 1449590567
create time : Thu Jul 07 16:24:54 ART 2005
last access : Thu Jul 07 16:41:30 ART 2005
max inactive interval : 1800
user name : anonymous
valid session : false
new session : true
overflowed : false
non-serializable app specific session data : {}
serializable app specific session data : {}
   at
com.ibm.ws.webcontainer.httpsession.SessionData.getValueGuts(SessionData.java(C
ompiled Code))
   at
com.ibm.ws.webcontainer.httpsession.SessionData.getValue(SessionData.java(Inlin
ed Compiled Code))
   at
com.ibm.ws.webcontainer.httpsession.SessionData.getAttribute(SessionData.java(I
nlined Compiled Code))
   at
com.ibm.ws.webcontainer.httpsession.HttpSessionFacade.getAttribute(HttpSessionF
acade.java(Compiled Code))
```

Check the servlet or JSP code that threw the exception to make sure the session is not being invalidated too early. If that is not the case, check the session timeout interval to ensure it is not too short.

Look at available online support

If these steps do not identify the problem, the following resources can be of help:

- Review the Java Servlet Specification Version 2.4, Section SRV.15.1.7 related to the HttpSession interface and methods definitions, to obtain more details for IllegalStateException creation causes:

  http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html

- Look up the extended error definitions for session and user profiles messages:

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.messages.doc/doc/SESN.html

For current information that is available from IBM Support on known problems and their resolution related to session management, use the following URL:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDS&rankprofile=8&dc=DB520+D800+D900+DA900+DA800&dtm

If these actions do not identify the problem, go to “The next step” on page 45 for information about gathering the MustGather documentation for session and session management problems.

Response generation problems

When an HTTP request from a client is delegated to a servlet, the service() method of the HttpServlet class is invoked. The HttpServlet class adds additional methods, such as doGet(), doPost(), doPut() and doHead(), for HTTP-based request processing. Figure 8 illustrates the life cycle of the service() method.

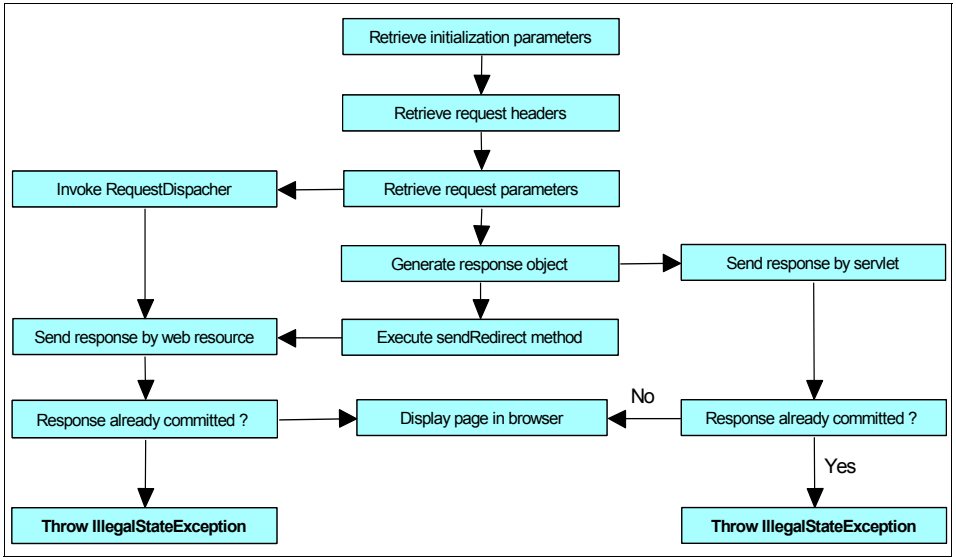

Figure 8 Servlet service() method life cycle

The `java.lang.IllegalStateException: Response already committed` exception is thrown by the HttpServletResponse interface during the response generation process (Example 8 on page 20). If the response has been committed, you cannot execute any method that is related to

HttpServletResponse object modification. For example, if you have written something in the response buffer, you cannot forward a page using the RequestDispacher interface methods.

Example 8 IllegalStateException in response generation

```
[7/8/05 20:36:25:694 ART] 0000004f ServletWrappe E   SRVE0068E: Could not
invoke the service() method on servlet TestServlet. Exception thrown :
java.lang.IllegalStateException
   at
com.ibm.ws.webcontainer.webapp.WebAppDispatcherContext.sendRedirect(WebAppDispa
tcherContext.java:486)
   at
com.ibm.ws.webcontainer.srt.SRTServletResponse.sendRedirect(SRTServletResponse.
java:810)
   at web.TestServlet.doGet(TestServlet.java:56)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:743)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:856)

[7/8/05 20:36:25:764 ART] 0000004f LocalTranCoor E   WLTC0017E: Resources
rolled back due to setRollbackOnly() being called.
[7/8/05 20:36:25:774 ART] 0000004f WebApp        E   SRVE0026E: [Servlet
Error]-[TestServlet]: java.lang.IllegalStateException
   at
com.ibm.ws.webcontainer.webapp.WebAppDispatcherContext.sendRedirect(WebAppDispa
tcherContext.java:486)
   at
com.ibm.ws.webcontainer.srt.SRTServletResponse.sendRedirect(SRTServletResponse.
java:810)
   at web.TestServlet.doGet(TestServlet.java:56)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:743)
   at javax.servlet.http.HttpServlet.service(HttpServlet.java:856)

[7/8/05 20:36:25:784 ART] 0000004f SRTServletRes W   WARNING: Cannot set
status. Response already committed.
```

Other issues to look for in an application that can cause a java.lang.IllegalStateException are the following calls when the response has already been committed:

- Calling setBufferSize()
- Calling ServletResponse.reset() or ServletResponse.resetBuffer()
- Calling either HttpServletResponse.sendError() or HttpServletResponse.sendRedirect().
- Calling RequestDispatcher.forward(), which includes performing a jsp:forward

Note: Remember that if you call forward() or sendRedirect() in your code, any lines of code following these will still execute.

Two common variants of this exception are:

- `java.lang.IllegalStateException: Header already sent`

  One or more headers have been committed to the client, so you cannot set that header again.

- `java.lang.IllegalStateException: Cannot forward as Output Stream or Writer has already been obtained`

  The calling servlet has called response.getWriter() or response.getOutputStream(). Because the response has been written, it is unsuitable for forwarding.

Look at available online support

If you still have not identified the cause of the problem, see the Java Servlet Specification Version 2.4 at the following URL to obtain more details for IllegalStateException generation causes in response generation process:

`http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html`

Review the following sections:

- SRV.14.2.5 RequestDispatcher interface
- SRV.14.2.16 ServletRequest interface
- SRV.14.2.22 ServletResponse interface

For current information that is available from IBM Support on known issues and resolutions that related to session management, go to the following URL:

`http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDF&rankprofile=8&dc=DB520+D800+D900+DA900+DA800&dtm`

If these steps do not resolve your problem, go to “The next step” on page 45 for information about gathering the MustGather documentation for servlet engine and Web container problems.

Analyzing problem areas

Your analysis of the data you gathered will most likely lead you to one of the following areas. If not, please see “The next step” on page 45.

Application URL specification

In a new installation or new application, it is possible that the URL of the application is not being specified correctly. To determine the URL of the installed application, you need to use the administrative console to view the configuration of a number of items.

The format of the URL is as follows:

`http://<host>:<port>/<context_root>/<resource_name>`

Finding *<host>:<port>*

When a Web module is installed, a virtual host is associated with it. To find the virtual host using the WebSphere administrative console:

1. Select **Applications** → **Enterprise Applications**.
2. Select the application name to open the details page.
3. In the Additional Properties section, click **Map virtual hosts for Web modules**.
4. Note the virtual host(s) that are used by the specified application Web module, as shown in Figure 9.

Figure 9 Listing the virtual host for a Web module

To find the host and port numbers that are valid for the virtual host:

1. Select **Environment** → **Virtual Hosts**.
2. Choose the virtual host, and then under Additional Properties, click **Host Aliases**.
3. The list contains the host name and port combinations that can be used to access this virtual host (Figure 10). The host column should contain values that are registered in a DNS server as a host name for the WebSphere server. An asterisk (*) in the host column indicates that any name can be used. In this case, use the server host name.

   If port 80 is listed, usually the request is being forwarded from a Web server. (The user specified the URL of the Web server, which is normally 80.)

   The WC_defaulthost (for example, 9080) and WC_defaulthost_secure (for example, 9443) ports for the application server should also be listed. You can see the corresponding ports by looking at the list of ports in the Communications section of the application server.

   Use either the host/port combination that accesses the virtual host through the Web server or the host/port combination for using WC_* ports to access the application directly through the application server.

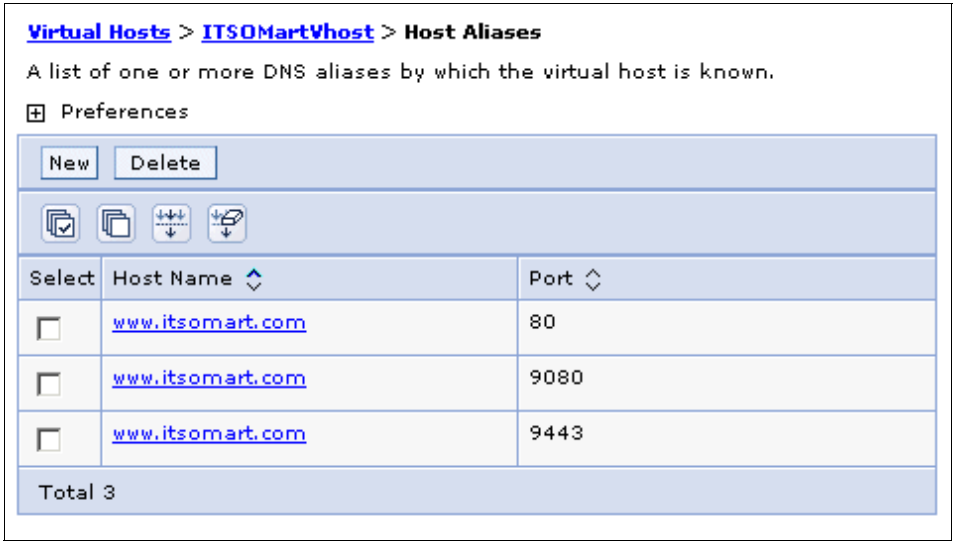

Figure 10 Finding the alias and ports for the virtual host

If you need SSL, you can check the Web container transport chains for the corresponding ports to see if SSL has been enabled by doing the following:

a. Select **Servers** → **Application servers**.

b. Select the application server name.

c. Under Container Settings, open the Web Container Settings list.

d. Click **Web container transport chains** and check the status and SSL enablement for the port that is specified in the virtual host alias.

Finding *<context_root>*

The context root is the Web application root and is used to identify the application in the URI. To find the context root:

1. In the WebSphere administrative console, click **Applications** → **Enterprise Applications**.
2. Select the target application, then under Additional Properties, click **View Deployment Descriptor**.
3. Review the *<context-root>* tag for the specified Web module in the related deployment descriptor tag (Figure 11).

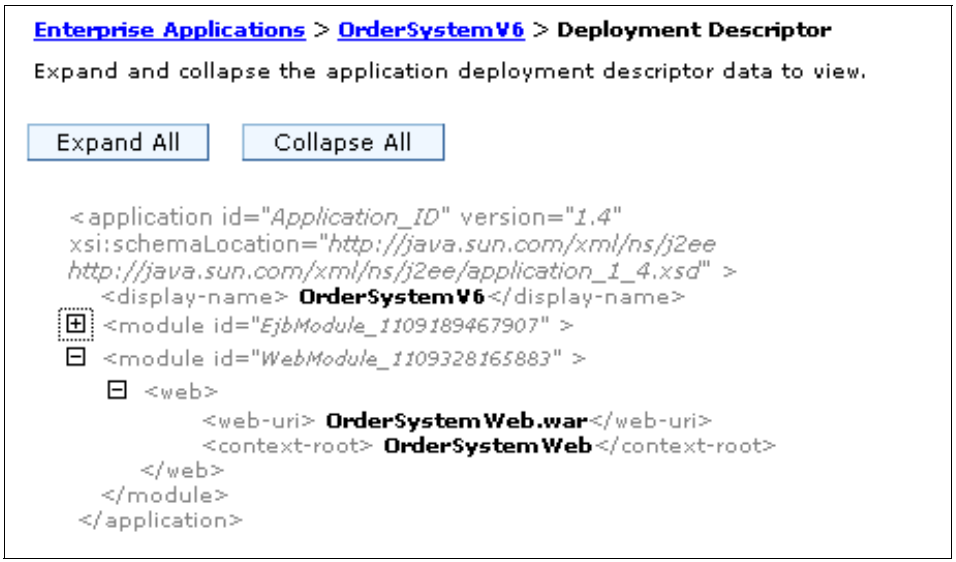

Figure 11 Finding the context root

Important: Remember that the Web container context root value "/" is used by the sample application named DefaultApplication. For this reason, if you need use this value and this sample application is installed, first uninstall the sample application, install your application, and regenerate the Web server plug-in.

Finding *<resource_name>*

To find the initial file for the application (for example, index.html), you can look at the welcome file list in the Web deployment descriptor. If you are looking for a particular servlet, the servlet mappings in the Web deployment descriptor tell you the URL pattern to use for each servlet. To find the initial file:

1. In the administrative console, click **Applications** → **Enterprise Applications**.
2. Select the target application, and then under Related Items, click **Web modules**.
3. Choose the specified Web module, and then under Additional Properties, click **View Deployment Descriptor**.
4. Examine the *<servlet-mapping id>* and *<welcome-file-list>* deployment descriptor tags to determine the URL pattern and file name configured respectively for this purpose (Example 9).

Example 9 Servlets and welcome files in Web deployment descriptor

```
<servlet-mapping id="ServletMapping_1" >
   <servlet-name>Main Servlet</servlet-name>
   <url-pattern>*.kht</url-pattern>
</servlet-mapping>
<servlet-mapping id="ServletMapping_2" >
   <servlet-name>Action Servlet</servlet-name>
   <url-pattern>*.do</url-pattern>
</servlet-mapping>
<servlet-mapping id="ServletMapping_3" >
   <servlet-name>Process Servlet</servlet-name>
   <url-pattern>/process</url-pattern>
</servlet-mapping>
<welcome-file-list id="WelcomeFileList_1" >
   <welcome-file> index.html</welcome-file>
   <welcome-file> index.htm</welcome-file>
</welcome-file-list>
```

5. If the Web module has security configured, also you need check the *<security-constraint>* and *<security-role>* deployment descriptor tags for the role that is needed for access to the selected Web resource (Example 10 on page 26).

Example 10 Security tags in Web deployment descriptor

```
<security-constraint id="SecurityConstraint_1" >
   <web-resource-collection id="WebResourceCollection_1" >
      <web-resource-name>Main Servlet</web-resource-name>
      <description>Access for Main Servlet.</description>
      <url-pattern>*.kht</url-pattern>
      <http-method>GET</http-method>
      <http-method>POST</http-method>
   </web-resource-collection>
   <auth-constraint id="AuthConstraint_1" >
      <description>Main Servlet Security</description>
      <role-name>All Role</role-name>
   </auth-constraint>
   <user-data-constraint id="UserDataConstraint_1" >
      <transport-guarantee>NONE</transport-guarantee>
   </user-data-constraint>
</security-constraint>
<security-role id="SecurityRole_1" >
   <description>All Authenticated Users Role.</description>
   <role-name>All Role</role-name>
</security-role>
```

Static resources not displayed

If the browser displays text output that is related to a JSP or servlet Web page but images or HTML files are not displayed, you could have a problem in the Web module packaging or in how the files are referenced within the application. Another possibility is that the file serving enablement feature needs to be turned on for your application.

Verify the static resource file locations

The first step is to verify that your files are in the right place and that the document root directory of the Web application module follows the J2EE standard, that is that the document root is in the *<web_module>*.war directory of the deployed application ear file. Typically this directory is in this location:

<WAS_install_root>/profiles/*<profile>*/installedApps/*<node>*/*<application>*.ear/*<web_module>*.war/

If the image files are in a subdirectory of the document root, verify that the reference to the image reflects that, as shown in Example 11.

Example 11 Image reference in HTML tags

```
File Location: <web_module_name.war>/images/test.gif
Correct HTML tag: <img SRC="images/test.gif">
Incorrect HTML tag: <img SRC="test.gif">
```

Note: Do not place files to be served to the client in the WEB-INF directory.

Check the file serving enablement feature

File serving allows a Web application to serve static file types (HTML, images, and style sheets) using the enable file servlet, also known as the file serving servlet or file serving enabler. This servlet serves up any resource file that is packaged in the WAR file, and the file serving enabled attribute is set to **true** by default. For more information about this feature, see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/cweb_flserv.html

This behavior is implemented by setting the `fileServingEnabled` property to `true` when configuring the Web module. If it is set to `false`, the Web server plug-in does not send requests for static content to the application server but leaves it up to the Web server to serve them. This is sometimes done for performance reasons to ensure that the content is served by the Web server. Serving the page from the Web server provides a shorter path to the page and usually provides more customization options than the file servlet can offer.

For more information, see:

- *Customizing SimpleFileServlet* to disable file serving at:

  http://www.ibm.com/support/docview.wss?uid=swg21116838

- Supported assembly tools in WebSphere Application Server V6:

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/catk_assemblytools.html

The `fileServingEnabled` property is located in the ibm-web-ext.xmi configuration file at:

<WAS_install_root>/profiles/*<profile>*/installedApps/*<node>*/*<application>*.ear\
<web_module>.war/WEB-INF/ibm-web-ext.xmi

To update the property using the Application Server Toolkit or a Rational® Developer tool:

1. Go to the J2EE Hierarchy view (Project Explorer in Rational), and select the target Web application module.
2. Double-click the Web deployment descriptor (web.xml), and select the Extensions tab to see the IBM Web module extensions (Figure 12 on page 28).
3. Under the General section, select or deselect **File serving enabled** to enable or disable the static file serving.

4. Save the Web deployment descriptor file.
5. Redeploy the Web module.
6. Regenerate the Web server plug-in, and propagate it to the Web server.
7. Stop and restart the Web server or allow enough time for the new plug-in configuration to be reloaded.
8. Retry the Web request.

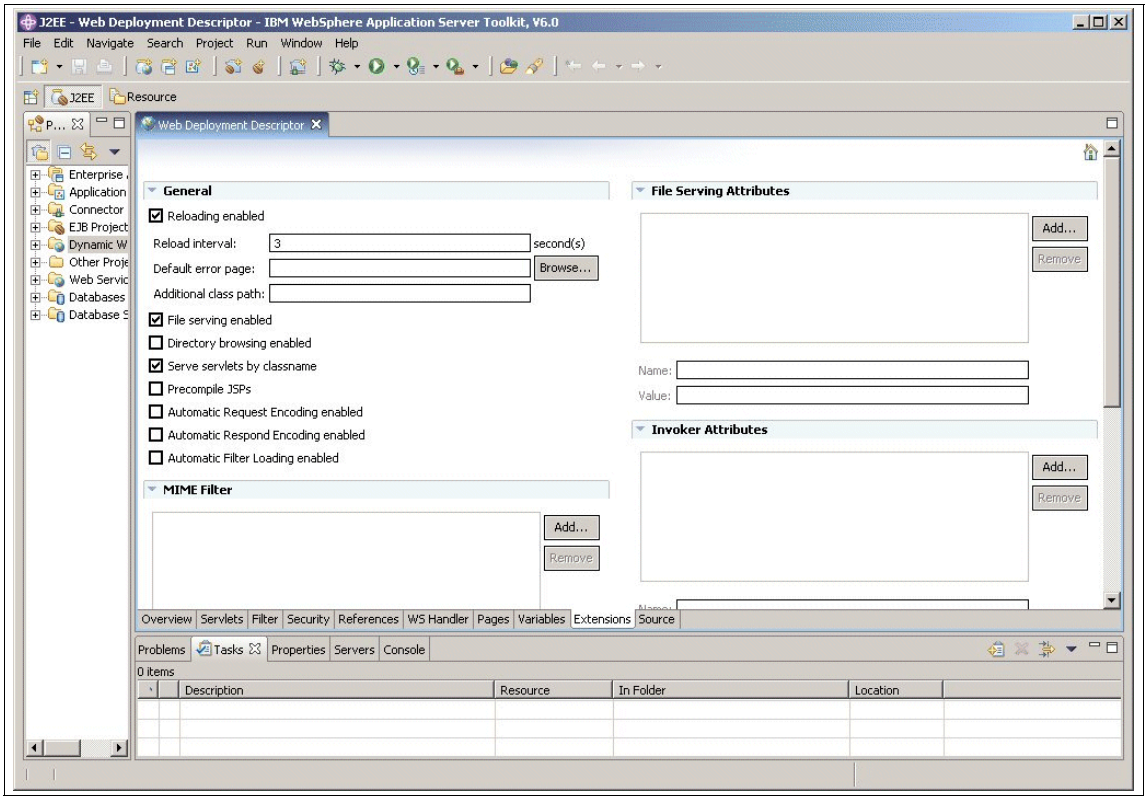

Figure 12 IBM Web module extensions

Web resources not reloading

If, after modifying and saving a servlet or JSP file, the change does not show up in the browser, you need to check the reload settings in the Web module configuration and the JSP runtime reload settings.

Web module reloading

For Web resources such as servlets and JSPs, the Web container reloads a Web module only when the IBM extension reloadingEnabled in the ibm-web-ext.xmi file is set to true. You can set reloadingEnabled to `true` when editing the Web module's extended deployment descriptors in an assembly tool.

To review these settings with the Application Server Toolkit or Rational Application Developer:

1. Go to the J2EE Hierarchy view (Project Explorer for Rational) and select the target Web application module.
2. Double-click the Web deployment descriptor (web.xml), and select the Extensions tab to see the IBM Web module extensions (Figure 12 on page 28).
3. In the General section, check the Reloading Enabled flag and the Reload Interval value.
4. Select **Reloading Enabled**, or if it is already selected, then set the **Reload Interval** lower.
5. Save the Web deployment descriptor file.
6. Redeploy the Web module.
7. Regenerate the Web server plug-in and propagate it to the Web server.
8. Stop and restart the Web server or wait for the new plug-in file to take effect.
9. Retry the Web request.

The entries in ibm-web-ext.xmi look similar to that shown in Example 12.

Example 12 Web module reloading settings in ibm-web-ext.xmi

```
<webappext:WebAppExtension xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:webappext="webappext.xmi" xmi:id="WebAppExtension_1109270589179"
reloadInterval="3" reloadingEnabled="true" fileServingEnabled="true">
```

Configuring JSP runtime reloading

You have the ability to modify the JSP processor behavior for different JSP stages (such as development, testing, or production environments). This done by configuring specific attributes in the IBM Web module extensions that affect the JSP runtime reload behavior.

JSP files can be translated and compiled at runtime when the JSP file or its dependencies are modified. This is known as JSP reloading. JSP reloading is enabled through the reloadEnabled JSP engine parameter in the ibm-web-ext.xmi configuration file.

To review these settings with the Application Server Toolkit or Rational Application Developer:

1. Go to the J2EE Hierarchy view (Project Explorer for Rational), and select the target Web application module.
2. Double-click in Web deployment descriptor (web.xml), and select the Extensions tab.
3. Under the JSP Attributes section (Figure 13), click **Add**.
4. Enter `reloadEnabled` in the Name field, `true` in the Value field, and click **Finish**.
5. Save the Web deployment descriptor file.
6. Redeploy the Web module.
7. Regenerate the Web server plug-in and verify it is working (see “Verify that the Web server plug-in is working” on page 9).
8. Stop and restart the Web server.
9. Retry the Web request.

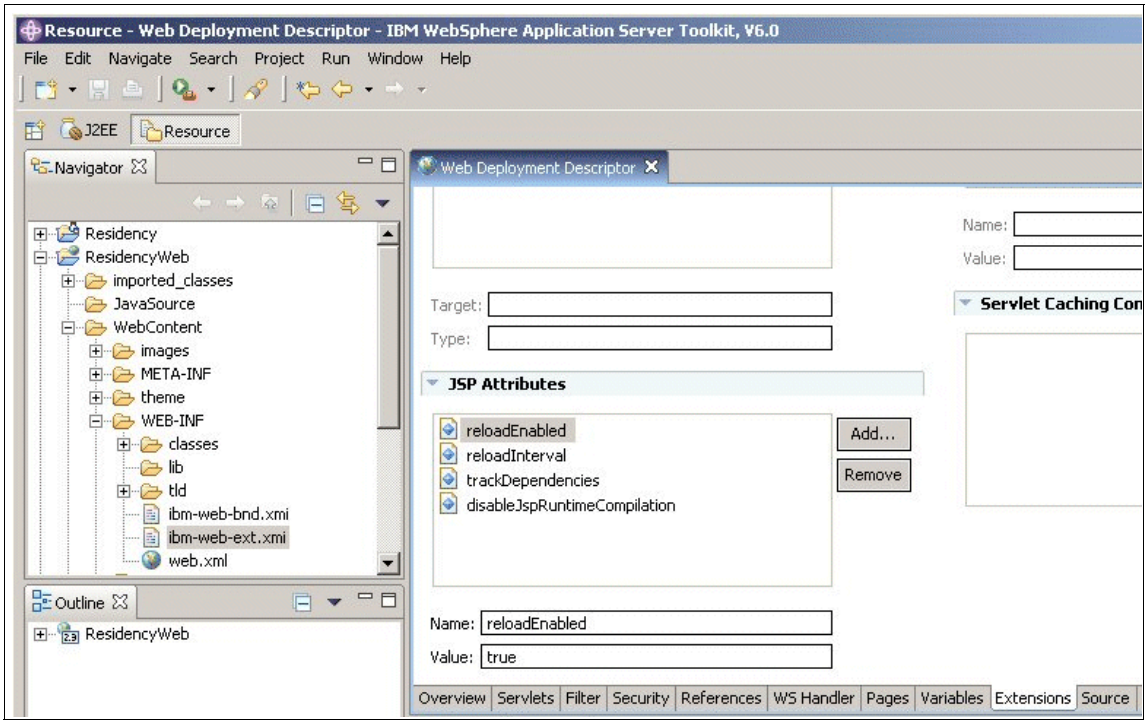

Figure 13 JSP attributes in IBM Web module extensions

The JSP engine settings are stored in ibm-web.ext.xmi, as shown in Example 13.

Example 13 JSP engine settings in ibm-web-ext.xmi

```
<jspAttributes xmi:id="JSPAttribute_1" name="reloadEnabled" value="true"/>
<jspAttributes xmi:id="JSPAttribute_9" name="reloadInterval" value="3"/>
```

For more available JSP attributes and details that are related to reload processing sequence, see the following article in the WebSphere Information Center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rweb_jspreloading.html

Encoding and internationalization issues

Encoding and internationalization problems usually surface as garbage characters in Web pages (Figure 14 on page 32). Another common symptom is that input from the user is interpreted incorrectly.

Double Byte Character Set (DBCS) character processing is a common issue for encoding problems. DBCS is used for languages such a Chinese, Korean, and Japanese, where a single byte is not sufficient to represent all characters in the alphabet. Proper coding and configuration usually resolves these problems

Figure 14 Web page with invalid character encoding settings

Character encoding in J2EE environment

Web components usually use java.io.PrintWriter object to produce responses; PrintWriter automatically encodes using ISO 8859-1 character encoding. Servlets can also output binary data using java.io.OutputStream objects, which perform no encoding. An application that cannot use the default encoding must explicitly set a different encoding.

For Web components, three encodings must be considered:

- Request
- JSP page
- Response

Request encoding

Request encoding is the character encoding in which parameters in an incoming request are interpreted. Currently, many browsers do not send a request encoding qualifier with the content-type HTTP header. In such cases, a Web container uses the default encoding: ISO-8859-1 to parse request data. If the client has not set character encoding and the request data is encoded with a different encoding from the default, the data will not be interpreted correctly.

To correct this situation, you can use the setCharacterEncoding(String enc) method to override the character encoding supplied by the container, as shown in Example 14.

Example 14 setCharacterEncoding() method implementation

```
public class TestServlet extends HttpServlet {

   public void doPost(HttpServletRequest req, HttpServletResponse resp)
      throws ServletException, IOException {

         req.setCharacterEncoding("UTF-8");
         String name = req.getParameter("name");
         resp.setContentType("text/html;charset=UTF-8");
         PrintWriter out = resp.getWriter();
         out.println("<html><body><h1>");
         out.println("Your name is "+name);
         out.println("</h1></body></html>");
   }
}
```

You must call the method before parsing any request parameters or reading any input from the request. Calling the method or tag after data has been read will not affect the encoding.

Page encoding

For JSP pages, page encoding is the character encoding in which the file is encoded. For JSP pages in standard syntax, the page encoding is determined from the following sources:

- The pageEncoding attribute of the page directive of the page
- The charset value of the contentType attribute of the page directive

If none of these is provided, ISO-8859-1 is used as the default page encoding.

Example 15 Page directive implementation

```
<HTML>
<HEAD>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="ISO-8859-15" %>

<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
<TITLE>Receiving parameters</TITLE>
</HEAD>
<BODY>
<H1>
Your data is <%= request.getParameter("name") %>
</H1>
</BODY>
</HTML>
```

The pageEncoding and contentType attributes determine the page character encoding of only the file that physically contains the page directive.

Response encoding

Response encoding is character encoding of the textual response that is generated from a Web component. A Web container sets an initial response encoding for a JSP page from the following sources:

- The charset value of the contentType attribute of the page directive
- The encoding specified by the pageEncoding attribute of the page directive

If none of these is provided, ISO-8859-1 is used as the default response encoding.

The setContentType() method in a servlet can be called to change the character encoding. Calls made after the getWriter() method has been called or after the response is committed have no effect on the character encoding.

MVC/Struts character encoding settings

When using a Model-View-Controller architecture, including the use of Struts, consider the following for encoding.

Request character encoding

In a Struts environment, call the setCharacterEncoding() method in the ActionForm, as shown in Example 16.

Example 16 Request encoding in Struts implementation

```
.
public class PostMessageForm extends ActionForm {
   public void reset(ActionMapping mapping, HttpServletRequest request) {
      try {
         request.setCharacterEncoding("UTF-8");
      }catch (UnsupportedEncodingException e) {
         e.printStackTrace();
      }
   }
}
.
```

Response character encoding

Specify the response encoding using the contentType attribute with a charset value in the JSP page directive.

WebSphere Application Server configuration

WebSphere determines the character encoding used for request/response by parsing the client input values in getParameter() and writing the output per the value in the accept-language header of the incoming request.

The language value and corresponding character encoding names are associated in the encoding.properties file, which is located in the *<WAS_install_root>/*properties directory (Example 17).

Example 17 encoding.properties file

```
.
en=ISO-8859-1
fr=ISO-8859-1
de=ISO-8859-1
es=ISO-8859-1
cs=ISO-8859-2
hr=ISO-8859-2
th=windows-874
vi=windows-1258
ja=Shift_JIS
ko=EUC-KR
zh=GB2312
zh_TW=Big5
hy=UTF-8
.
```

There might be cases where you want to override the definition in encoding.properties. For example, when you want to use UTF-8 for the entire application server, use the JVM command line argument **client.encoding.override** for the selected application server.

Specify **-Dclient.encoding.override=UTF-8** for the generic JVM arguments field in the Java Virtual Machine section.

For more information, see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/trun_svr_utf.html

autoRequestEncoding and autoResponseEncoding

Starting with WebSphere Application Server Version 5, the Web container no longer sets request and response encodings and response content types automatically. The default value for both extensions is false, then the request and response character encoding is set to the Servlet 2.4 Specification default, which is ISO-8859-1.

Use an assembly tool (Figure 12 on page 28) to change the default values for the autoRequestEncoding and autoResponseEncoding extensions.

Review the autoRequestEncoding and autoResponseEncoding encoding examples for a description of Web container behavior when these values are set to true:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rweb_autoreq.html

Summary

Figure 15 shows a summary of how the encoding values are determined.

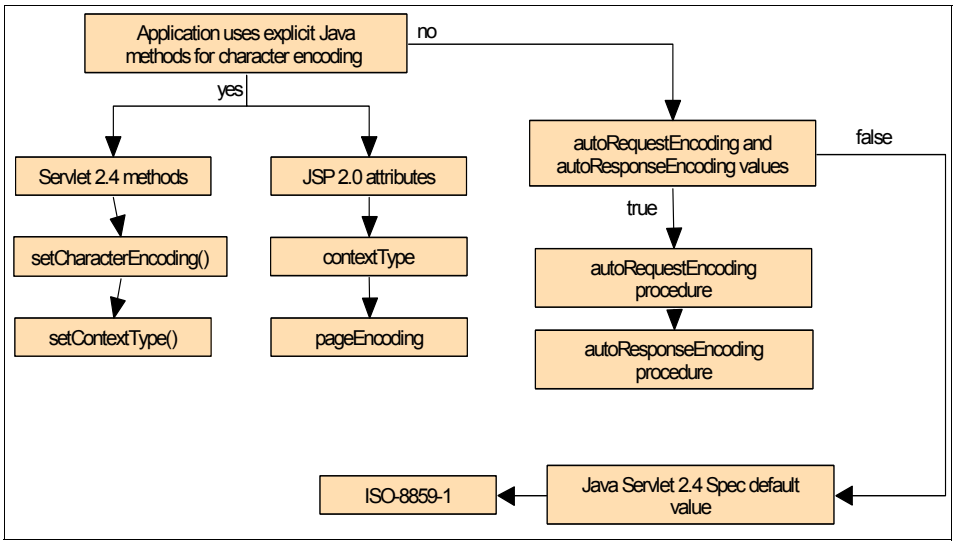

Figure 15 Encoding determination process

Look at available online support

If none of these steps fixes your problem, the following resources might be helpful:

- *Globalize your On Demand Business* for tips on character encoding:

  `http://www-306.ibm.com/software/globalization/j2ee/encoding.jsp`

- The Java Servlet Specification Version 2.4 for details about the character encoding available methods:

  `http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html`

- The JavaServer Pages Specification Version2.0, section JSP.4 that are related to internationalization issues:

  `http://jcp.org/aboutJava/communityprocess/final/jsr152/index.html`

- *Internationalization: Resources for learning*

  `http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rin_resources.html`

For current information available from IBM Support on known issues and resolutions that are related to encoding, see:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCVRZX&rankprofile=8&dc=DB520+D800+D900+DA900+DA800&dtm

If these steps do not resolve your problem, go to “The next step” on page 45 for information about gathering the MustGather documentation for internationalization problems.

HTTP session management

Figure 16 outlines the problem determination steps for HTTP session management problems.

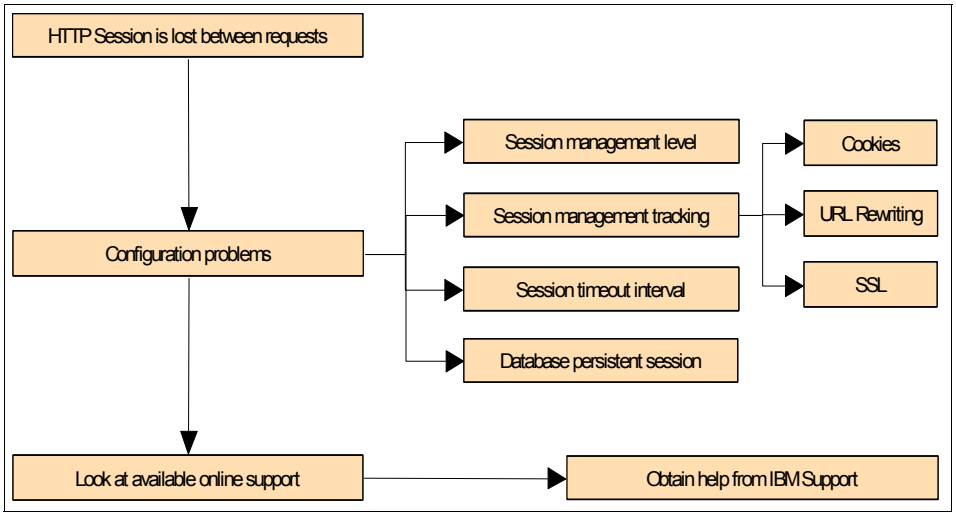

Figure 16 HTTP Session problems steps

When session data is lost between requests, the session manager configuration settings are the most likely suspect.

IBMTrackerDebug:

A special servlet, com.ibm.ws.webcontainer.httpsession.IBMTrackerDebug, can be invoked to display the current configuration and statistics related to session tracking. This servlet has all the counters that are in performance monitor tool and has some additional counters.

It can be invoked from any Web module that is enabled to serve servlets by class name using the following URL:

`http://localhost:9080/servlet/com.ibm.ws.webcontainer.httpsession.IBMTrackerDebug`

Note that a module that can be viewed via the `serve servlets by classname` feature can be seen by anyone that can view the application. You might want to map a specific, secured URL to the servlet instead and disable `serve servlets by classname`.

Session management settings can be set at the following levels:

- Web container
- Enterprise application
- Web module

When you configure session management at the Web container level, all applications and the respective Web modules in the Web container normally inherit that configuration, setting up a basic default configuration for the applications and Web modules below it. However, you can set up different configurations individually for specific applications and Web modules that vary from the Web container default. These different configurations override the default for these applications and Web modules only.

Note: When you overwrite the default session management settings on the application level, all the Web modules below that application inherit this new setting, unless they too are set to overwrite these settings.

To begin session management configuration, you first need to get to the settings for the selected level.

1. Web container level:
   a. In the WebSphere administrative console, click **Servers** → **Application servers**.
   b. Select the application server name.

c. Under Container Settings, expand the Web Container Settings list, and click **Session management**.

2. Enterprise application level:
   a. In the WebSphere administrative console, click **Applications** → **Enterprise Applications**.
   b. Click the application name, and under Additional Properties, click **Session management**.
3. Web module level:
   a. In the WebSphere administrative console, click **Applications** → **Enterprise Applications**.
   b. Select the target application, and then under Related Items, click **Web modules**.
   c. Choose the specified Web module, and then under Additional Properties, click **Session management**.

Note: If you are working on the Web module or application level and want these settings to override the inherited Session Management settings, under General Properties, select **Override session management**.

WebSphere Application Server offers the following methods of session manager tracking:

- Session management tracking with cookies
- Session management tracking with URL rewriting
- Session management tracking with SSL

If you are having session management problems, there are specific things to check for each type of session manager tracking. In addition, the following configuration items are common to all three tracking types:

- Session timeout interval
- Database persistent session

Session management tracking with cookies

The session manager uses cookies to store the session ID on the client between requests. To check the session management cookie settings, go to the appropriate session management configuration level and ensure that the `Enable cookies` option is set.

To view or change the cookie settings, click the **Enable cookies** link.

To ensure that cookies are flowing between the application server and the browser:

- Make sure cookies are enabled on the browser you are testing from or from which your users are accessing the application.
- Check the cookie flow between the browser and server. On the browser, enable cookie prompt. Enter the URL of the servlet and make sure the cookie is being prompted.
- If you use a Netscape browser, check the following URL for information related to cookie domains that are not recognized with Netscape:

  http://www.ibm.com/support/docview.wss?uid=swg21163880

  Check the cookie domain definition (see Example 18).

Example 18 Cookie domain definition

```
Cookie domain: ".myCom.com"
Resources should be accessed using that domain name:
http://www.myCom.com/myapp/servlet/sessionservlet
```

- Check the cookie path that is specified in the session manager settings. Check whether the problem URL is hierarchically below the cookie path that is specified. If it is not, correct the cookie path.
- If the `cookie maximum age` property is set, ensure that the client (browser) machine's date and time is the same as the server's, including the time zone. If the client and the server time difference is over the maximum age, every access would be a new session because the cookie will expire after the access.
- If you have multiple Web modules within an enterprise application that track sessions:
  - If you want to have different session settings among Web modules in an enterprise application, ensure that each Web module specifies a different cookie name or path.
  - If Web modules within an enterprise application use a common cookie name and path, ensure that the HTTP session settings, such as `cookie maximum age`, are the same for all Web modules. Otherwise, cookie behavior is unpredictable and depends upon which application creates the session. Note that this does not affect session data, which is maintained separately by the Web module.
- Configure Internet browsers to present a warning before accepting cookies:

  http://www.ibm.com/support/docview.wss?uid=swg21157880

Session management tracking with URL rewriting

Using URL rewriting for session management means that the session management facility uses rewritten URLs to carry the session IDs. If URL rewriting is enabled, the session management facility recognizes session IDs that arrive in the URL if the encodeURL method is called in the servlet.

Enabling protocol switch rewriting specifies that the session ID is added to a URL when the URL requires a switch from HTTP to HTTPS or from HTTPS to HTTP. If rewriting is enabled, the session ID is required to go between HTTP and HTTPS.

You can check if URL rewriting is enabled by going to the appropriate session management configuration level and verifying that the `Enable URL rewriting` option is selected. You can also see if `Enable protocol switch rewriting` option is selected.

If you are using URL rewriting:

- Ensure that there are no static HTML pages on the application's navigation path.
- Review that the servlet and JSP files are implementing URL rewriting correctly. For details and an example see:

  `http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rprs_sesd.html`
- If the application uses Struts, check this URL to understand the URL rewrite behavior in WebSphere Application Server V6:

  `http://www.ibm.com/support/docview.wss?uid=swg21205259`

Session management tracking with SSL

No special programming is required to track sessions with Secure Sockets Layer (SSL) information. You can check to see if SSL session management is enabled by going to the appropriate session management configuration level and verifying that the `Enable SSL ID tracking` option is selected.

- The SSL session ID is negotiated between the Web browser and Web server. This ID cannot survive a Web server failure. However, the failure of an application server does not affect the SSL session ID if an external Web server is present between WebSphere Application Server and the browser.
- SSL tracking is supported for the IBM HTTP Server and iPlanet™ Web servers only.
- You can control the lifetime of an SSL session ID by configuring options in the Web server. For example, in the IBM HTTP Server, set the configuration variable SSLV3TIMEOUT to provide an adequate lifetime for the SSL session

ID. An interval that is too short can cause a premature termination of a session.

Session timeout interval

The session manager invalidation process thread runs every *x* seconds to invalidate any invalid sessions, where *x* is determined based on the `session timeout interval` that is specified in the session manager properties. For the default value of 30 minutes, *x* is around 300 seconds. In this case, it could take up to 5 minutes (300 seconds) beyond the timeout threshold of 30 minutes for a particular session to become invalidated.

Database persistent session

If session data is lost when the application server restarts or is not shared across the cluster, you might have a problem with the database used to persist sessions or with its configuration.

To review the database session persistent values using the administrative console, do the following:

1. In the session management settings, under the Additional Properties list, click **Distributed environment settings**.
2. If you are using a database for persistent sessions, the Database option is selected.
3. Click the **Database** link to check the data source definition.
4. Check the JNDI name of the data source.
5. Ensure that the correct user ID and password that are required to access the database are specified.
6. The data source should be non-JTA, that is non-XA enabled.
7. With DB2®, for row sizes other than 4 KB, make sure that the specified row size matches the DB2 page size. Make sure the tablespace name is specified correctly.
8. Check the SystemOut.log file for appropriate database error messages.

Note: You should check these settings against the properties of an existing data source. The session manager does not create a session database automatically for you.

For more information, see:

- *Session manager user ID settings for database persistence*

  http://www.ibm.com/support/docview.wss?uid=swg21203466

- *Creating a DB2 table for session persistence - WebSphere V6*

  http://www.ibm.com/support/docview.wss?uid=swg21199312

Look at available online support

If none of these steps fixes your problem, the following resources might be helpful:

- *Managing HTTP sessions: Resources for learning*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rprs_r4ln.html

- *Best practices for using HTTP Sessions*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/cprs_sesm.html

- *Tuning session management*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/cprs_sesm.html

- *State replication in the Web tier*

  http://www.ibm.com/developerworks/java/library/j-jtp07294.html

For current information available from IBM Support on known issues and resolutions that are related to session management, see:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDS&rankprofile=8&dc=DB520+D800+D900+DA900+DA800&dtm

If these steps do not resolve your problem, go to the next section for information about gathering the MustGather documentation for session management problems.

The next step

The symptoms and problem areas included in this paper are some that you are more likely to experience. However, there are other things that can go wrong, or the cause of the problem might be related to a component other than the Web container.

If, after going through this process, you still have an undiagnosed problem, it is recommended that you go back to *Approach to Problem Determination in WebSphere Application Server V6* at:

http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf

Review the problem classifications to see if there are any other components that might be causing the problem.

If you feel sure that you have a Web container related problem, there are things you can do before contacting IBM support. First, you should review the documentation that you have gathered for errors related to the Web container that were not addressed in this paper and search support sites for information or fixes. For hints and tips, technotes, and fixes that are related to Web container problems, see:

http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPDF

If the problem is performance related, the following references are useful for configuring the JSP engine for optimal performance:

- *JSP engine configuration parameters*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rweb_jspengine.html

- *JSP class file generation*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/cweb_jspclassfiles.html

- *Disabling JavaServer Pages run-time compilation*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rweb_jspdis.html

- *Packages and directories for generated .java and .class files*

  http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/cweb_javapkg.html

If this search does not provide any information relevant to your problem, it might be time to contact IBM support. The MustGather documents for Web container problems are:

- HTTP status code 404, “NOT FOUND” in WebSphere Application Server V6.0

  http://www.ibm.com/support/docview.wss?uid=swg21193551

- *MustGather: Servlet engine and Web container errors on WebSphere Application Server V6.0*

  http://www.ibm.com/support/docview.wss?uid=swg21193538

- *MustGather: JavaServer Pages JSP exceptions for V6.0*

  http://www.ibm.com/support/docview.wss?uid=swg21193099

- *MustGather: Sessions and session management problems in V6.0*

  http://www.ibm.com/support/docview.wss?uid=swg21192604

- *MustGather: i18n (Internationalization) / Double Byte Character Set (DBCS)*

  http://www-1.ibm.com/support/docview.wss?rs=180&uid=swg21141732

Be sure to note all of the diagnostic work that you have done so far to minimize the amount of time that it takes IBM Support to assist you in resolving your problem.

You can find detailed information about contacting IBM support for issues that are related to WebSphere Application Server V6.0 in *Approach to Problem Determination in WebSphere Application Server V6* at:

http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

This document created or updated on September 30, 2005.

Send us your comments in one of the following ways:

- Use the online **Contact us** review redbook form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbook@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HZ8 Building 662, P.O. Box 12195
  Research Triangle Park, NC 27709-2195 U.S.A.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

@server®	Domino®	Rational®
@server®	DB2®	Redbooks™
Redbooks (logo) ™	IBM®	WebSphere®

The following terms are trademarks of other companies:

iPlanet, Java, JavaServer, JavaServer Pages, JSP, JVM, J2EE, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.