Debugging J2EE Applications

Release 640

HELP.BCJAVA_START_QUICK

Copyright

Icons in Body Text

Icon	Meaning
⚠	Caution
💬	Example
💡	Note
🧭	Recommendation
syn	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help* → *General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Typographic Conventions

Type Style	Description
Example text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation.
`Example text`	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example text`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, `F2` or `ENTER`.

Debugging J2EE Applications

The SAP NetWeaver Developer Studio provides support when debugging your Web applications. By making special settings for the server process of the J2EE Engine and setting breakpoints, you can interrupt the execution of your Web application and go through it step by step.

In this tutorial, you will be familiarized with the SAP NetWeaver Developer Studio settings required for debugging. You will then debug a JSP and a servlet from the car rental application [Extern].

Next step:

Activating Debugging [Seite 5]

Activating Debugging

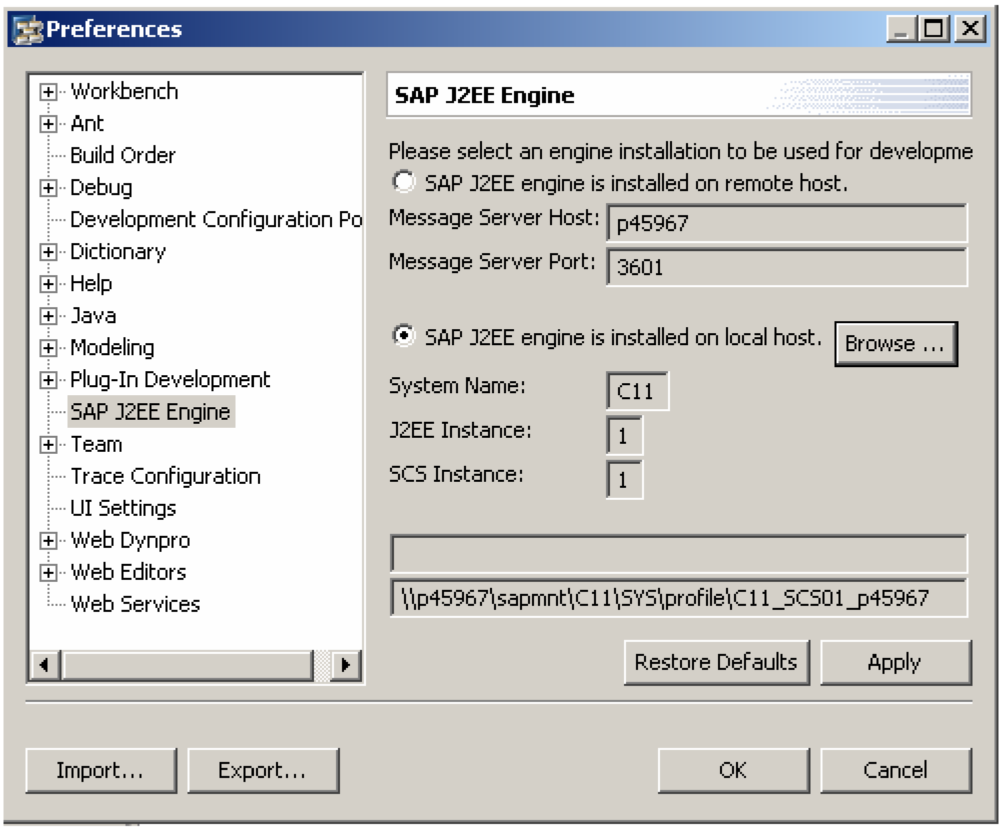
You need at first to switch to the debug mode on the J2EE Engine before you can debug your J2EE application (Servlets, JSP's, …) or a Web Dynpro application. You do this step from within the Developer Studio.

Prerequisites

- You have launched the Developer Studio.

Procedure

1. Check the relevant settings by choosing *Window -> Preferences*.
2. Choose the node *SAP J2EE Engine*.
3. If not already done so, specify the appropiate local J2EE Engine installation by choosing the *Browse…* button.



SAP J2EE Engine

Please select an engine installation to be used for developme

SAP J2EE engine is installed on remote host.

Message Server Host: p45967

Message Server Port: 3601

[✓ SAP J2EE engine is installed on local host.]

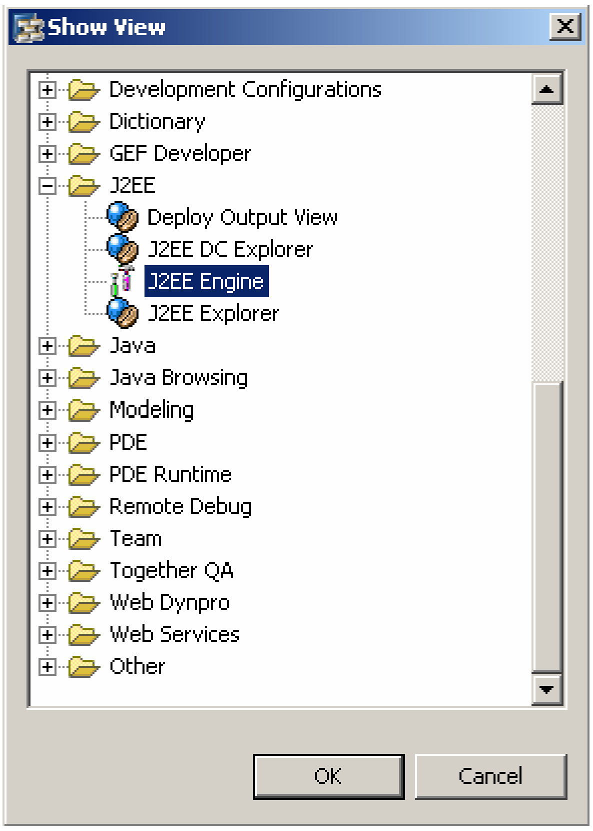
System Name: C11

J2EE Instance: 1

SCS Instance: 1

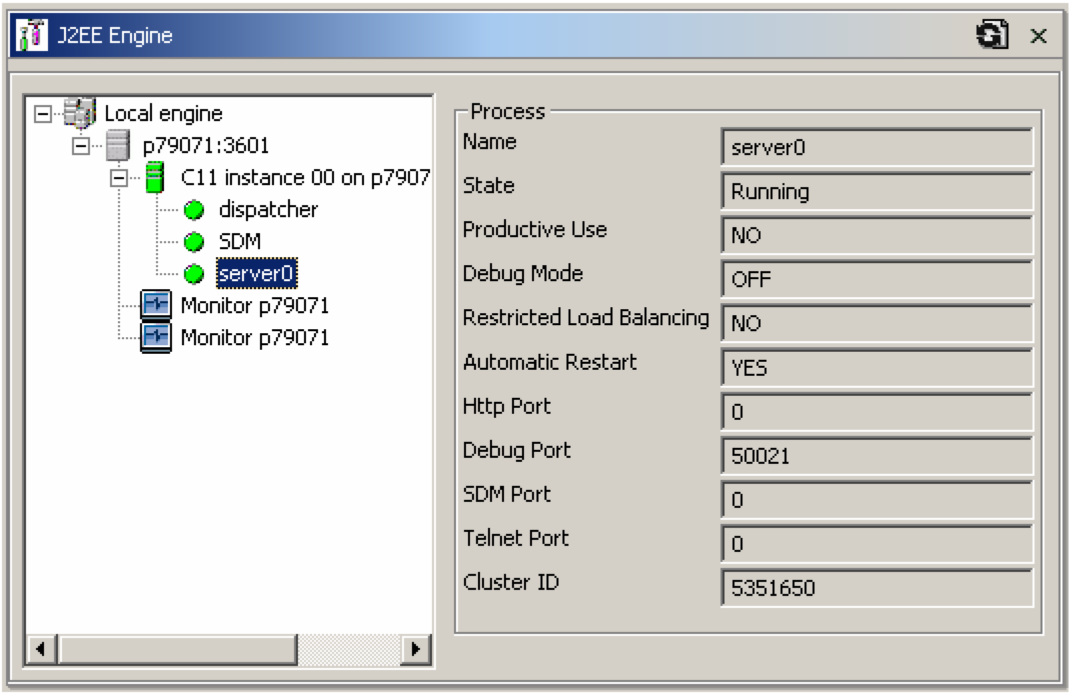
\\p45967\sapmnt\C11\SYS\profile\C11_SCS01_p45967

4. To be able to debug within a running J2EE application, you must activate debugging for the server process of the J2EE Engine. You activate this in the *J2EE Engine* view.
   To do so, choose *Window -> Show View -> Other*. Select *J2EE -> J2EE Engine* and choose *OK*.



The system displays a view containing status information about the running J2EE Engine.

5. Expand the tree display fully until you can see the actual server process *server0*.

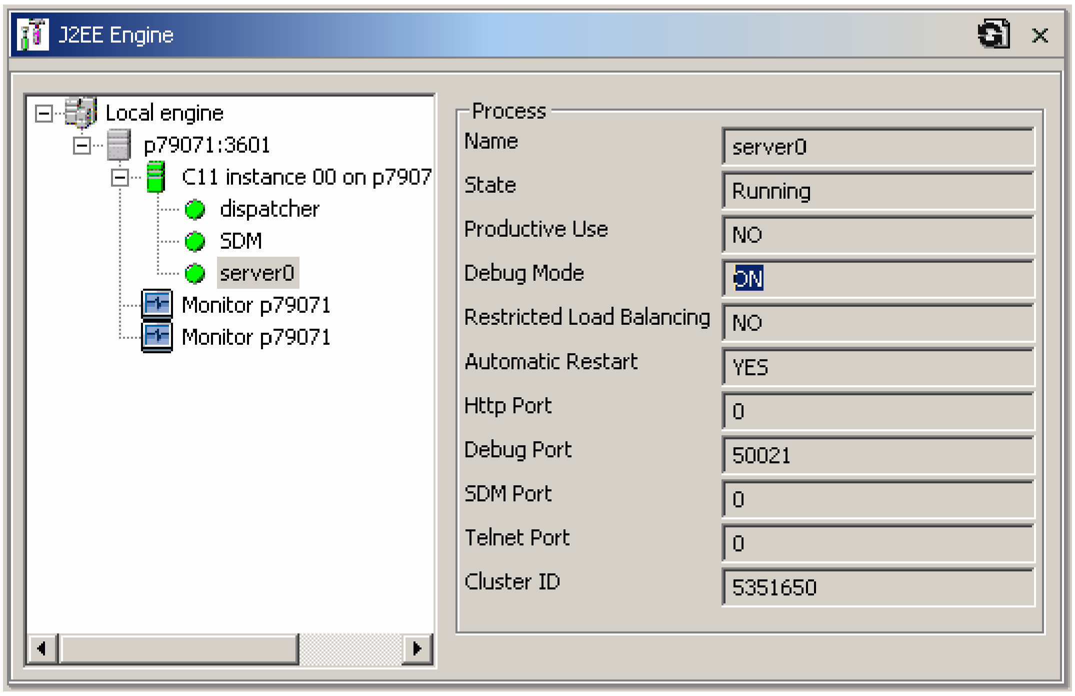


Name	server0
State	Running
Productive Use	NO
Debug Mode	OFF
Restricted Load Balancing	NO
Automatic Restart	YES
Http Port	0
Debug Port	50021
SDM Port	0
Telnet Port	0
Cluster ID	5351650

6. Right-click the *server0* node and choose *Enable debugging of process* from the context menu.

Result

The server process is stopped and restarted in debugging mode. To see the current state of the server, choose the *Refresh tree* icon in the view toolbar. Wait until the *Debug Mode* changes to *On*.



Name	server0
State	Running
Productive Use	NO
Debug Mode	ON
Restricted Load Balancing	NO
Automatic Restart	YES
Http Port	0
Debug Port	50021
SDM Port	0
Telnet Port	0
Cluster ID	5351650

Next step:

Preparations for Debugging a JSP [Seite 8]

Preparations for Debugging a JSP

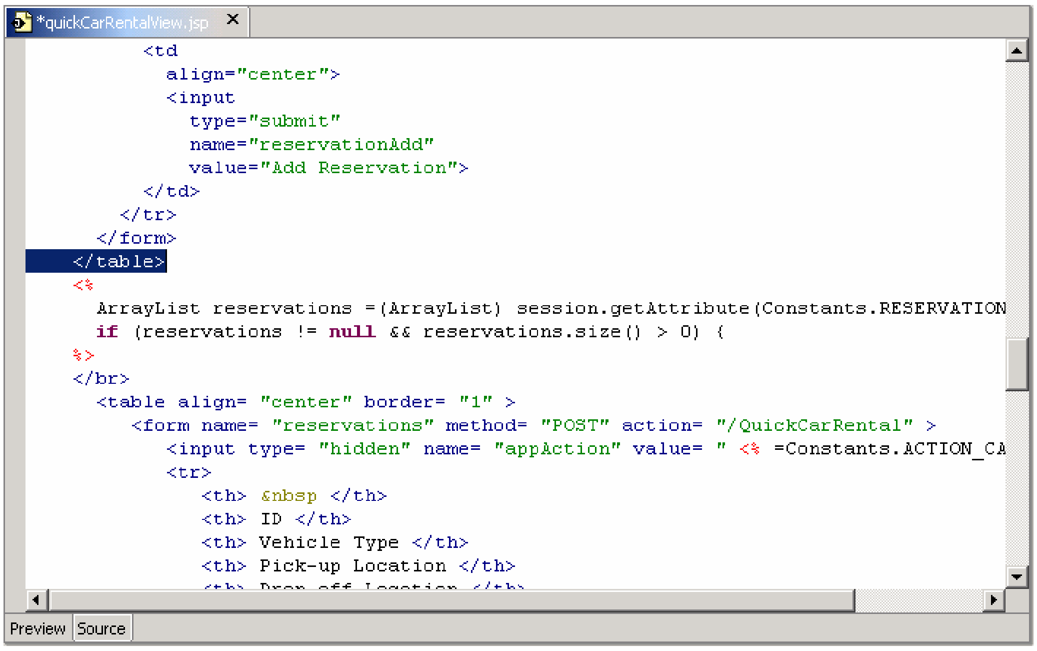
In the following scenario, you will change the source code of the JSP *quickCarRentalView.jsp*, rebuild the Web application, and deploy it so you can start debugging.

Prerequisites

- The Software Deployment Manager (SDM) must be launched before deploying the Web application.
- You have the car rental project opened in SAP NetWeaver Developer Studio

Procedure

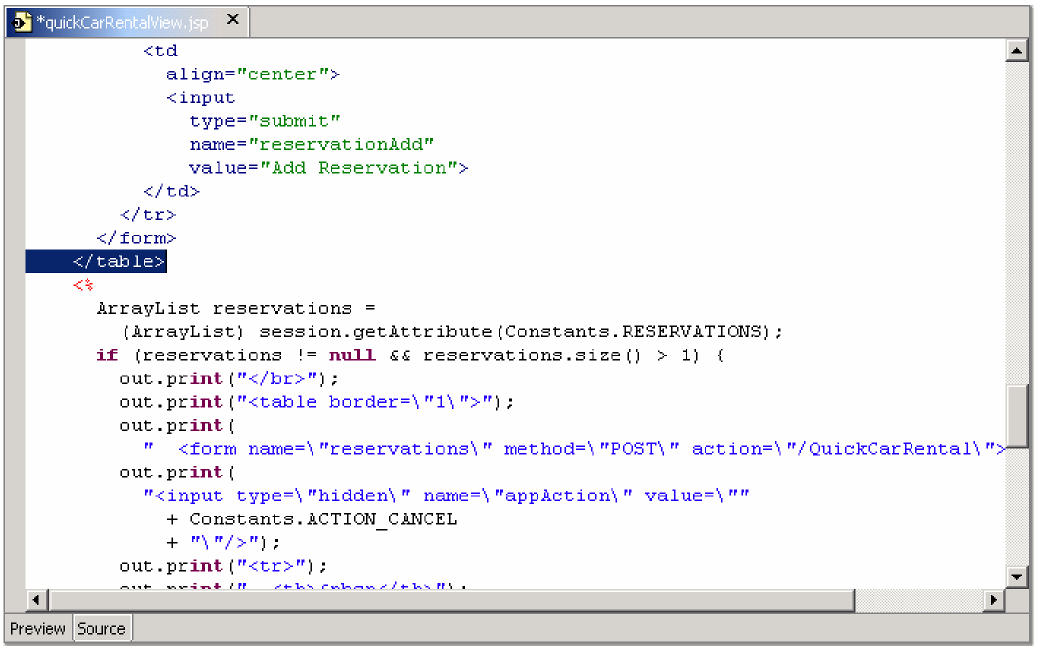
1. Open the source code of the JSP *QuickCarRentalView.jsp*: In the *J2EE Explorer*, double-click *J2EE_QuickCarRentalWeb* → *Web content* → *quickCarRentalView.jsp*
2. Display the source code and navigate to the line at which the reservations are output in tabular form (at the end of the JSP).



*quickCarRentalView.jsp

```
          <td
            align="center">
            <input
              type="submit"
              name="reservationAdd"
              value="Add Reservation">
          </td>
        </tr>
      </form>
    </table>
    <%
      ArrayList reservations =(ArrayList) session.getAttribute(Constants.RESERVATION
      if (reservations != null && reservations.size() > 0) {
    %>
    </br>
      <table align= "center" border= "1" >
         <form name= "reservations" method= "POST" action= "/QuickCarRental" >
            <input type= "hidden" name= "appAction" value= " <% =Constants.ACTION_CA
            <tr>
               <th> &nbsp </th>
               <th> ID </th>
               <th> Vehicle Type </th>
               <th> Pick-up Location </th>
               <th> Drop-off Location </th>
```

3. Change the (HTML) statements (till the end of the file).



*quickCarRentalView.jsp

```
          <td
            align="center">
            <input
              type="submit"
              name="reservationAdd"
              value="Add Reservation">
          </td>
        </tr>
      </form>
    </table>
    <%
      ArrayList reservations =
        (ArrayList) session.getAttribute(Constants.RESERVATIONS);
      if (reservations != null && reservations.size() > 1) {
        out.print("</br>");
        out.print("<table border=\"1\">");
        out.print(
          "   <form name=\"reservations\" method=\"POST\" action=\"/QuickCarRental\">
        out.print(
          "<input type=\"hidden\" name=\"appAction\" value=\""
            + Constants.ACTION_CANCEL
            + "\"/>");
        out.print("<tr>");
        out.print("    <th>&nbsp</th>");
```

Use the following code:

```
<% ArrayList reservations = (ArrayList) session.getAttribute(Constants.RESERVATIONS);
  if (reservations != null && reservations.size() > 0) {
    out.print("</br>");
    out.print("<table border=\"1\">");
    out.print("<form name=\"reservations\" method=\"POST\" action=\"/QuickCarRental\">");
    out.print("<input type=\"hidden\" name=\"appAction\" value=\"" + Constants.ACTION_CANCEL + "\"/>");
    out.print("<tr>");
    out.print("  <th>&nbsp</th>");
    out.print("  <th>ID</th>");
    out.print("  <th>Vehicle Type</th>");
```

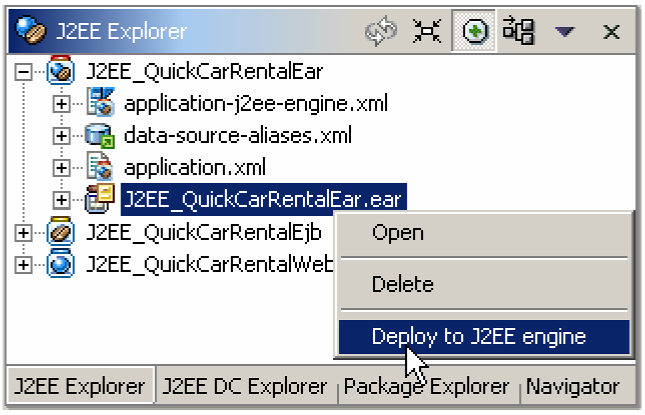
```
     out.print("  <th>Pick-up Location</th>");
     out.print("  <th>Drop-off Location</th>");
     out.print("  <th>Begin Date</th>");
     out.print("  <th>End Date</th>");
     out.print("  <th>Price</th>");
     out.print("</tr>");
     for (int i = 0; i < reservations.size(); i++) {
     String[] reservation = (String[]) reservations.get(i);
     out.print("<tr>");
     out.print(" <td>");
     out.print(" <input type=\"checkbox\" name=\"check\"  value=\"" + reservation[0] + "\"/>");
     out.print("  </td>");
     for (int j = 0; j < reservation.length; j++) {
     String value = reservation[j];
     out.print("<td>" + value + "</td>");
     }
     out.print("</tr>");
     }
     out.print("<tr>");
     out.print("<td align=\"center\" colspan=\"8\">");
     out.print("<table>");
     out.print("<tr>");
     out.print("  <td align=\"left\">");
     out.print("    <input type=\"submit\" name=\"reservationCancel\" value=\"Cancel Reservation\">");
     out.print("  </td>");
     out.print("  <td/>");
     out.print("</tr>");
     out.print("</form>");
     out.print("</table>");
     } else {
     out.print("No Reservations");
     }
%>

</body>
</html>
```

4. Save the change by choosing the appropriate icon in the toolbar.
5. Choose *Project -> Rebuild All* from the menu.
6. Create an EAR file.

   In the J2EE Explorer, right-click the node *J2EE_QuickCarRentalEar* and choose *Build EAR File* from the context menu.
7. Deploy the EAR file.

   In the J2EE Explorer, right-click the node *J2EE_QuickCarRentalEar* → *J2EE_QuickCarRentalEar.ear* and choose *Deploy to J2EE engine* from the context menu.



8. After the successful deployment start the Web application using the corresponding URL:

   `http://localhost:50000/QuickCarRental`

Result

The change has no effect on the application itself. However, the replacement of the HTML statements with the Java codes enables you to improve the debugging of the JSP page in the next step.

Next step:

Debugging a JSP [Seite 11]

Debugging a JSP

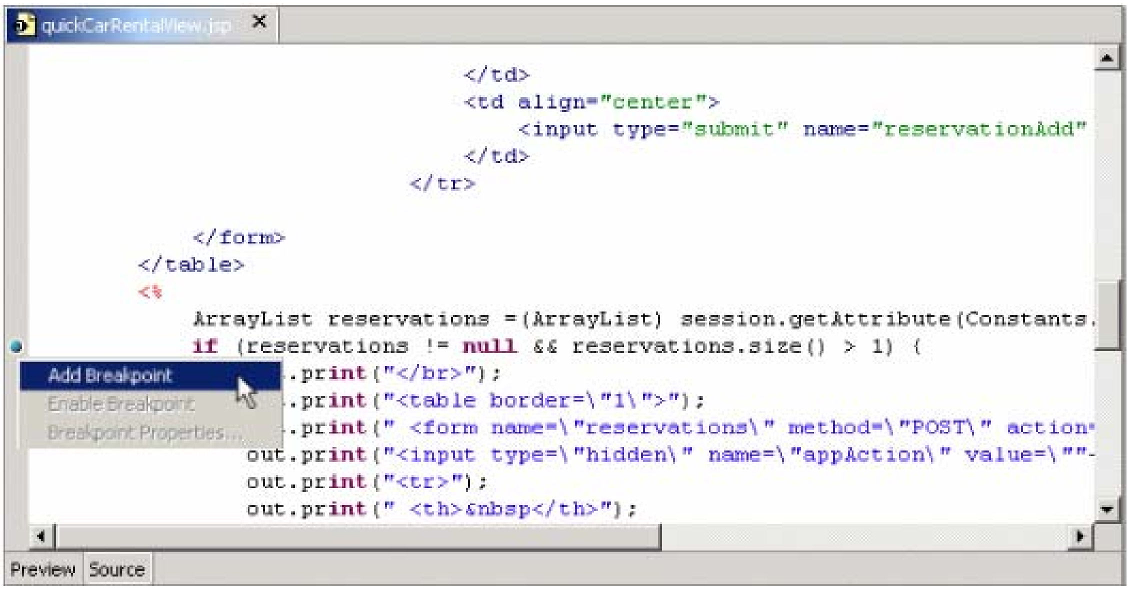
The following scenario describes how you can interrupt the execution of a JSP by adding a breakpoint to analyze the state of the page during runtime.

Prerequisites

☐ Debugging is activated – that is, the server process has been stopped and restarted in debugging mode.

Procedure

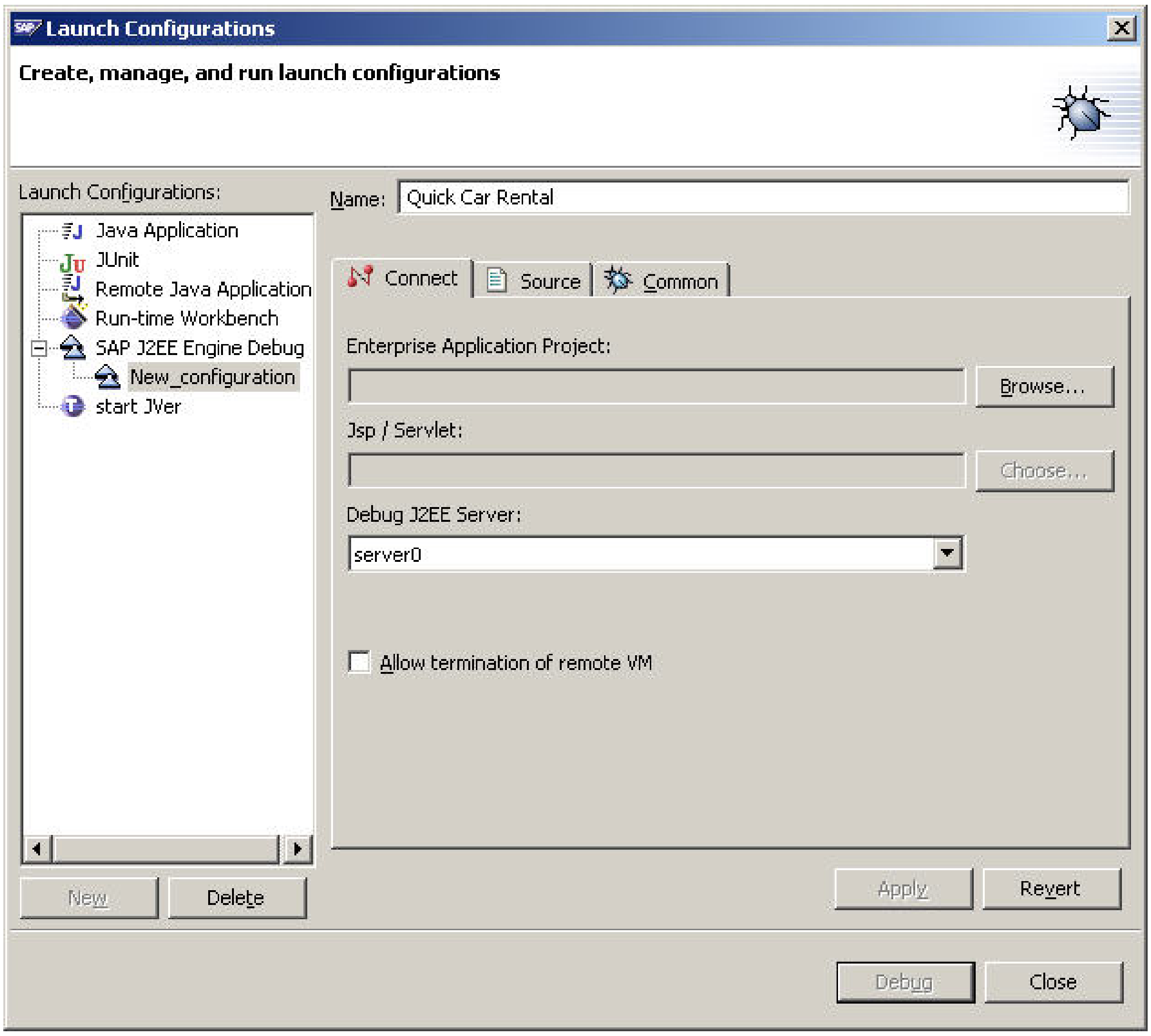
1. Open the source code of the JSP *quickCarRentalView.jsp*: In the *J2EE Explorer*, double-click *J2EE_QuickCarRentalWeb* → *webContent* → *quickCarRentalView.jsp.*
2. Display the source code and navigate to the beginning of the loop through the existing reservations. Right-click the left bar of the editor frame above the appropriate line to open the context menu and choose *Add Breakpoint*.



```
                                </td>
                                <td align="center">
                                    <input type="submit" name="reservationAdd"
                                </td>
                            </tr>

                </form>
            </table>
            <%
                ArrayList reservations =(ArrayList) session.getAttribute(Constants.
                if (reservations != null && reservations.size() > 1) {
                    .print("</br>");
                    .print("<table border=\"1\">");
                    .print(" <form name=\"reservations\" method=\"POST\" action=
                    out.print("<input type=\"hidden\" name=\"appAction\" value=\""-
                    out.print("<tr>");
                    out.print(" <th>&nbsp;</th>");
```

3. To start the J2EE application in the debugger, you require a *launch configuration*.
   Choose *Run* → *Debug...* in the main menu.
   In the list, select *J2EE Application* and choose *New*.



Launch Configurations

Create, manage, and run launch configurations

Launch Configurations:
- Java Application
- JUnit
- Remote Java Application
- Run-time Workbench
- SAP J2EE Engine Debug
  - New_configuration
- start JVer

Name: Quick Car Rental

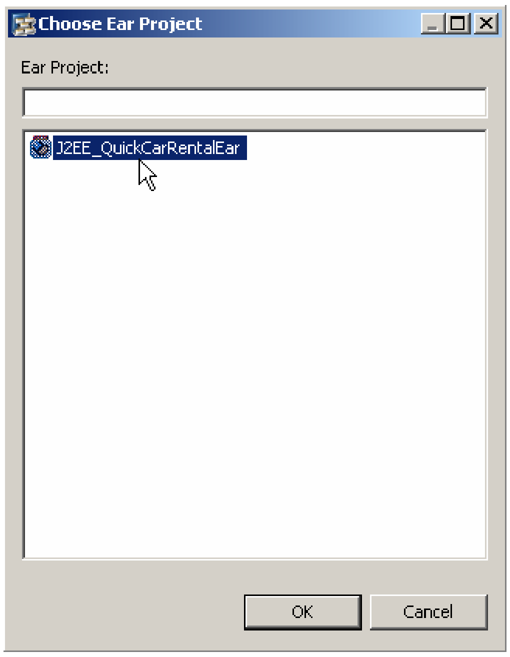
Enterprise Application Project:

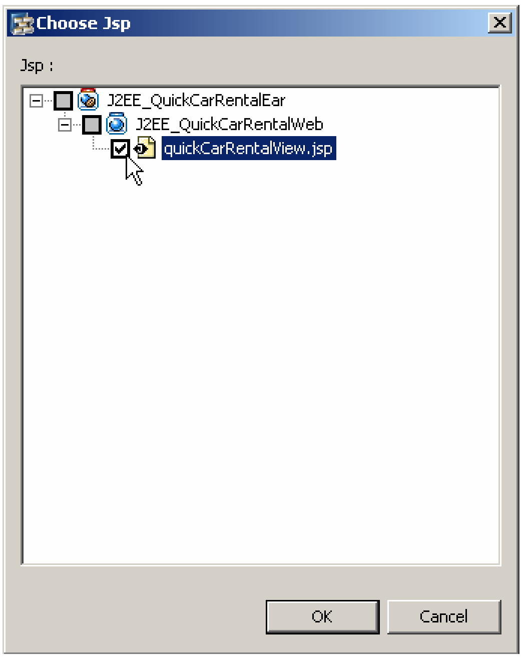
Jsp / Servlet:

Debug J2EE Server: server0

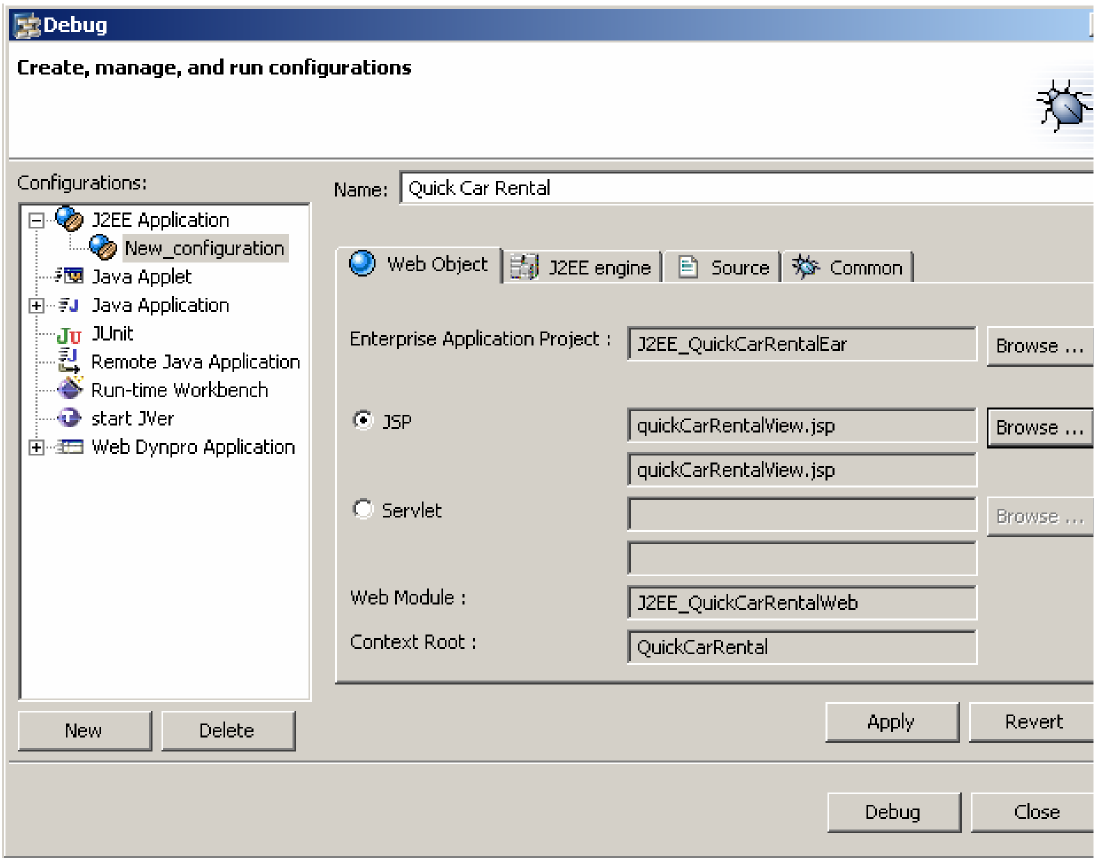
☐ Allow termination of remote VM

4. In the *Name* field, enter **`Quick Car Rental`**.
5. Choose the *Browse…* button next to the field *Enterprise Application Project*, select the project *J2EE_QuickCarRentalEar*, and choose *OK*.
6. Choose the RadioButton *JSP* and click the *Browse…* button next to the field *Jsp*, select the JSP *quickCarRentalView.jsp*, and choose *OK*.





7. The configuration is now complete; you can start the debugger by choosing *Debug*.



Name: Quick Car Rental

Enterprise Application Project : J2EE_QuickCarRentalEar

(●) JSP quickCarRentalView.jsp

quickCarRentalView.jsp

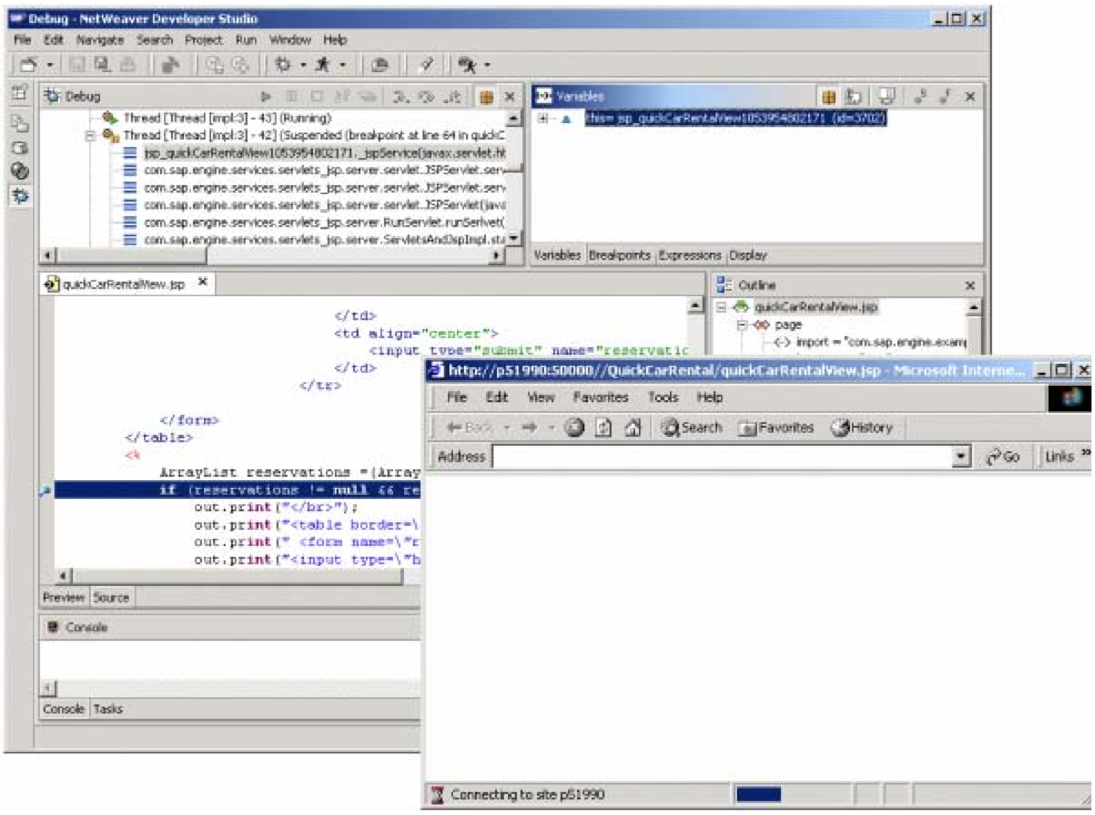
( ) Servlet

Web Module : J2EE_QuickCarRentalWeb

Context Root : QuickCarRental

Result

The SAP NetWeaver Studio automatically switches to the debug perspective. The Web application is started in an external browser and seems to hang. However, if you change back to the SAP NetWeaver Developer Studio, you will see that the application was stopped at the breakpoint and can now be analyzed.


Debug - NetWeaver Developer Studio
File Edit Navigate Search Project Run Window Help
Debug
Variables
Variables Breakpoints Expressions Display
quickCarRentalView.jsp
Outline
Preview Source
Console
Console Tasks
File Edit View Favorites Tools Help
Back Search Favorites History
Address
Go
Links

```
Thread [Thread [impl:3] - 43] (Running)
Thread [Thread [impl:3] - 42] (Suspended (breakpoint at line 64 in quickC
    jsp_quickCarRentalView1053954802171._jspService(javax.servlet.ht
    com.sap.engine.services.servlets_jsp.server.servlet.JSPServlet.serv
    com.sap.engine.services.servlets_jsp.server.servlet.JSPServlet.serv
    com.sap.engine.services.servlets_jsp.server.servlet.JSPServlet(java
    com.sap.engine.services.servlets_jsp.server.RunServlet.runServlet(
    com.sap.engine.services.servlets_jsp.server.ServletsAndJspImpl.sta
```

```
this= jsp_quickCarRentalView1053954802171 (id=3702)
```

```
                                </td>
                                <td align="center">
                                    <input type="submit" name="reservatic
                                </td>
                            </tr>

                </form>
            </table>
            <%
                ArrayList reservations =(Array
                if (reservations != null && re
                    out.print("</br>");
                    out.print("<table border=\
                    out.print(" <form name=\"r
                    out.print("<input type=\"b
```

```
quickCarRentalView.jsp
    page
        import = "com.sap.engine.exam
```

```
http://p51990:50000//QuickCarRental/quickCarRentalView.jsp - Microsoft Interne
Connecting to site p51990
```

You have the following options:

Key	Description
F5: *Step Into*	Jumps to the next statement
F6: *Step Over*	The next command is executed without jumping to the current statement
F7: *Step Return*	If you previously chose F5, you can choose F7 to cancel the debugging of the current command

Terminating Debugging

If you want to exit debugging, you must terminate the threads in the IDE.

Proceed as follows:

1. In the *Debug* window, select the top node (*QuickCarRental[J2EE Application]*).
2. Right-click and choose *Terminate* from the context menu.
3. Then choose *Remove All Terminated* from the context menu.

Next step: Preparations for Debugging a Servlet [Seite 15]

Preparations for Debugging a Servlet

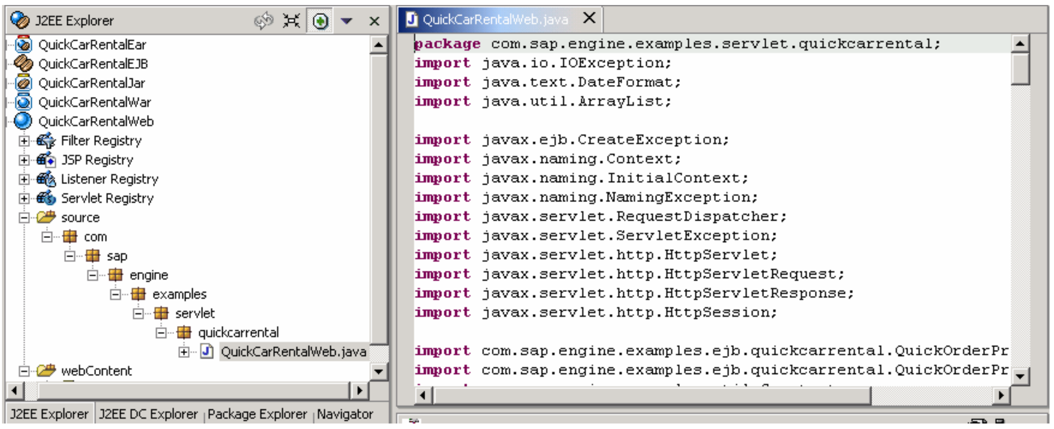
In the following scenario, you will incorporate a deliberate error in a servlet, rebuild the Web application, and deploy it so you can start debugging.

Prerequisites

- The Software Deployment Manager (SDM) must be launched before deploying the Web application.

Procedure

1. Open the source code of the servlet *QuickReservationServlet.java* In the *J2EE Explorer*, double-click *J2EE_QuickCarRentalWeb* → *source* → **...** → *QuickReservationServlet.java*
2. Display the source code and navigate to the `saveAction` method.



```
package com.sap.engine.examples.servlet.quickcarrental;
import java.io.IOException;
import java.text.DateFormat;
import java.util.ArrayList;

import javax.ejb.CreateException;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.sap.engine.examples.ejb.quickcarrental.QuickOrderPr
import com.sap.engine.examples.ejb.quickcarrental.QuickOrderPr
```

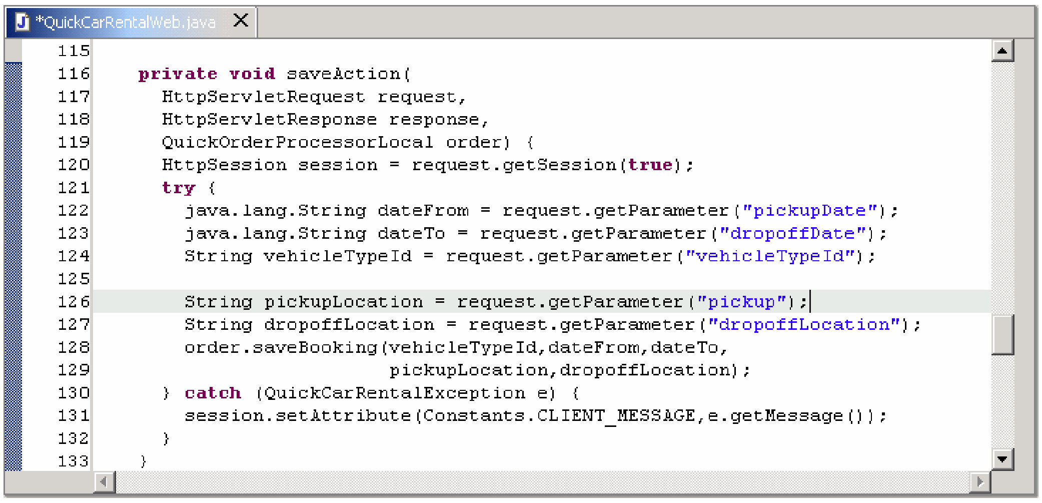
3. Change the statement:

```
String pickupLocation = request.getParameter("pickupLocation");
```

as follows:

```
String pickupLocation = request.getParameter("pickup");
```

As a result of this simple typing error, no valid value is assigned to the `pickupLocation` variable and the program terminates with an exception.


*QuickCarRentalWeb.java

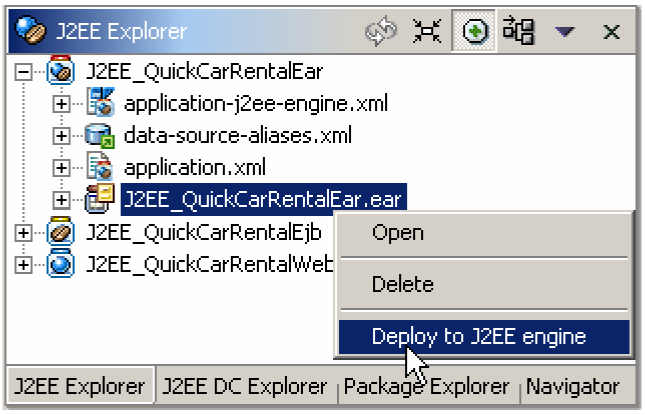
```
  private void saveAction(
    HttpServletRequest request,
    HttpServletResponse response,
    QuickOrderProcessorLocal order) {
    HttpSession session = request.getSession(true);
    try {
      java.lang.String dateFrom = request.getParameter("pickupDate");
      java.lang.String dateTo = request.getParameter("dropoffDate");
      String vehicleTypeId = request.getParameter("vehicleTypeId");

      String pickupLocation = request.getParameter("pickup");
      String dropoffLocation = request.getParameter("dropoffLocation");
      order.saveBooking(vehicleTypeId,dateFrom,dateTo,
                        pickupLocation,dropoffLocation);
    } catch (QuickCarRentalException e) {
      session.setAttribute(Constants.CLIENT_MESSAGE,e.getMessage());
    }
  }
```

4. Save the change.
5. Choose *Project -> Rebuild All* from the menu.
6. Create an EAR file.

   In the J2EE Explorer, right-click the project node *J2EE_QuickCarRentalEar* and choose *Build EAR File* from the context menu.
7. Deploy the EAR file.

   In the J2EE Explorer, right-click the node *J2EE_QuickCarRentalEar* → *J2EE_QuickCarRentalEar.ear* and choose *Deploy to J2EE engine* from the context menu. (Note: The SDM must be launched for a successful deployment.)

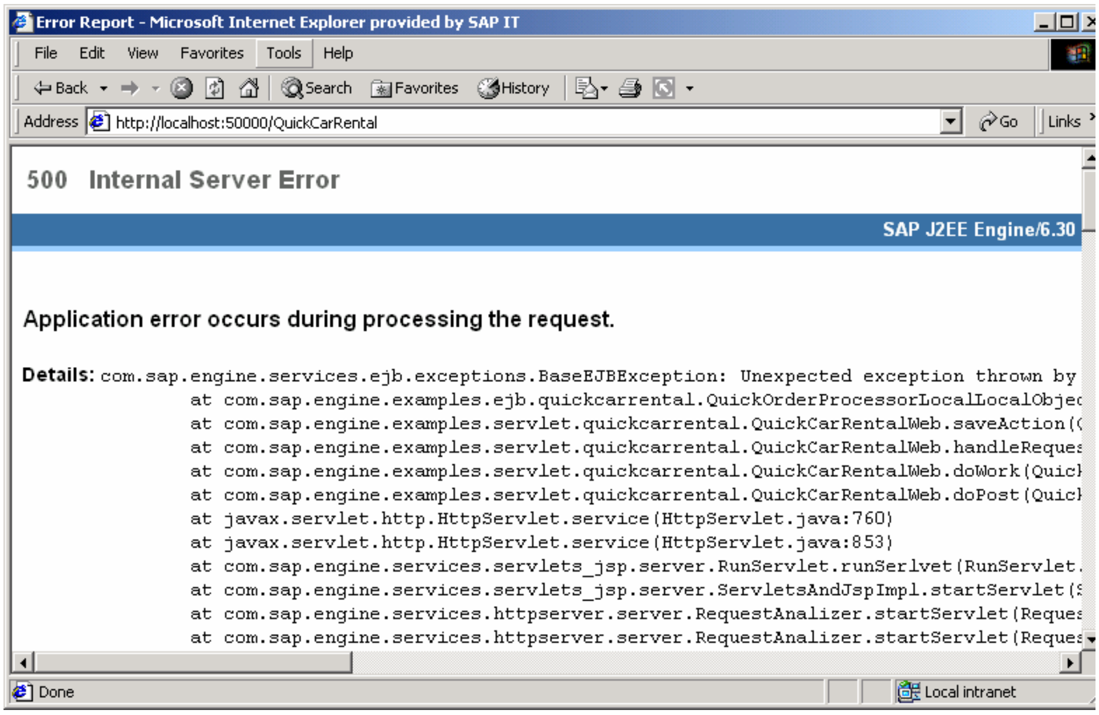


8. After the successful deployment start the Web application using the following URL:

   `http://localhost:50000/QuickCarRental`

Result

An error message occurs during the execution of the application when you try to store a reservation using the button *Add Reservation*.


Error Report - Microsoft Internet Explorer provided by SAP IT
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost:50000/QuickCarRental
Go Links
Done
Local intranet

500 Internal Server Error

SAP J2EE Engine/6.30

Application error occurs during processing the request.

```
Details: com.sap.engine.services.ejb.exceptions.BaseEJBException: Unexpected exception thrown by
              at com.sap.engine.examples.ejb.quickcarrental.QuickOrderProcessorLocalLocalObjec
              at com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.saveAction(
              at com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.handleReques
              at com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.doWork(Quick
              at com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.doPost(Quick
              at javax.servlet.http.HttpServlet.service(HttpServlet.java:760)
              at javax.servlet.http.HttpServlet.service(HttpServlet.java:853)
              at com.sap.engine.services.servlets_jsp.server.RunServlet.runSerlvet(RunServlet.
              at com.sap.engine.services.servlets_jsp.server.ServletsAndJspImpl.startServlet(S
              at com.sap.engine.services.httpserver.server.RequestAnalizer.startServlet(Reques
              at com.sap.engine.services.httpserver.server.RequestAnalizer.startServlet(Reques
```

This means that the error occurs during the call of the `saveBooking` method, which is called in the servlet method `saveAction`.

Next step:

Debugging a Servlet [Seite 17]

Debugging a Servlet

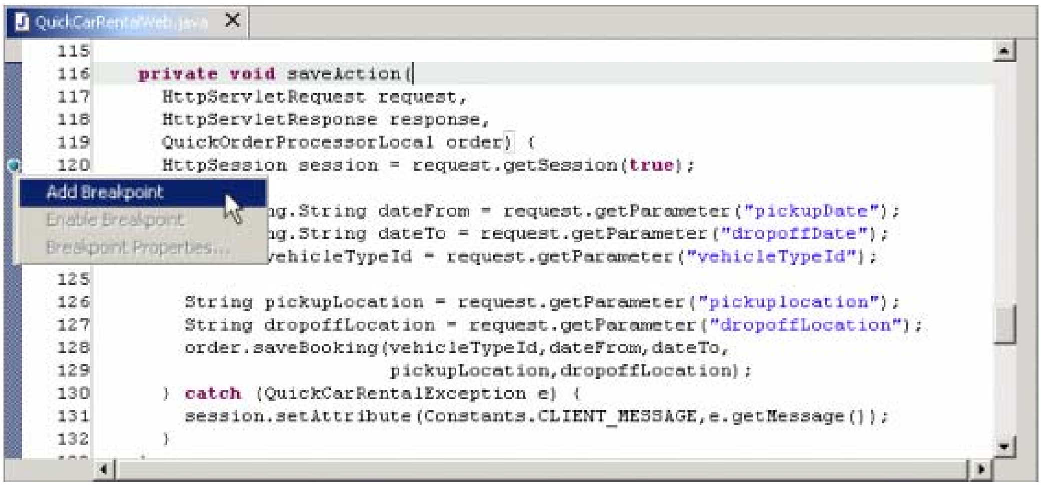
An error occurred during the execution of your application and you want to analyze it.

Prerequisites

☐ Debugging is activated – that is, the server process has been stopped and restarted in debugging mode.

Procedure

1. Open the source code of the servlet *QuickReservationServlet.java*: In the *J2EE Explorer*, double-click *J2EE_QuickCarRentalWeb* → *source* → *com* → **...** → *QuickReservationServlet.java*
2. Display the source code and navigate to the beginning of the `saveAction` method.
3. In the `saveAction` method, right-click the left bar of the editor frame above the first Java statement to open the context menu and choose *Add Breakpoint*.

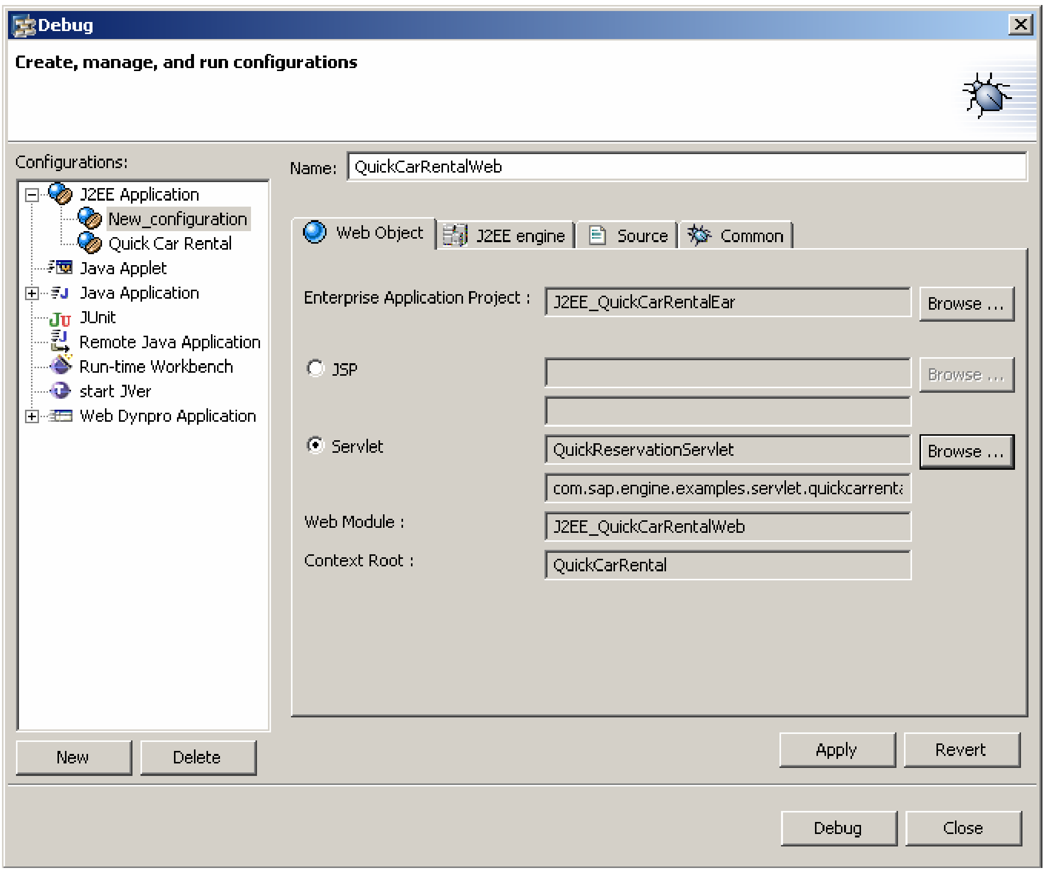


```
    private void saveAction(
      HttpServletRequest request,
      HttpServletResponse response,
      QuickOrderProcessorLocal order) {
      HttpSession session = request.getSession(true);

        ng.String dateFrom = request.getParameter("pickupDate");
        ng.String dateTo = request.getParameter("dropoffDate");
        vehicleTypeId = request.getParameter("vehicleTypeId");

        String pickupLocation = request.getParameter("pickuplocation");
        String dropoffLocation = request.getParameter("dropoffLocation");
        order.saveBooking(vehicleTypeId,dateFrom,dateTo,
                          pickupLocation,dropoffLocation);
      } catch (QuickCarRentalException e) {
        session.setAttribute(Constants.CLIENT_MESSAGE,e.getMessage());
      }
```

4. To start the J2EE application in the debugger, you require a *launch configuration*.
   Choose *Run* → *Debug...* in the main menu.
   In the list, select *J2EE Application* and choose *New*.



Name: QuickCarRentalWeb

Enterprise Application Project : J2EE_QuickCarRentalEar

JSP

[✓ Servlet] QuickReservationServlet

com.sap.engine.examples.servlet.quickcarrent

Web Module : J2EE_QuickCarRentalWeb

Context Root : QuickCarRental

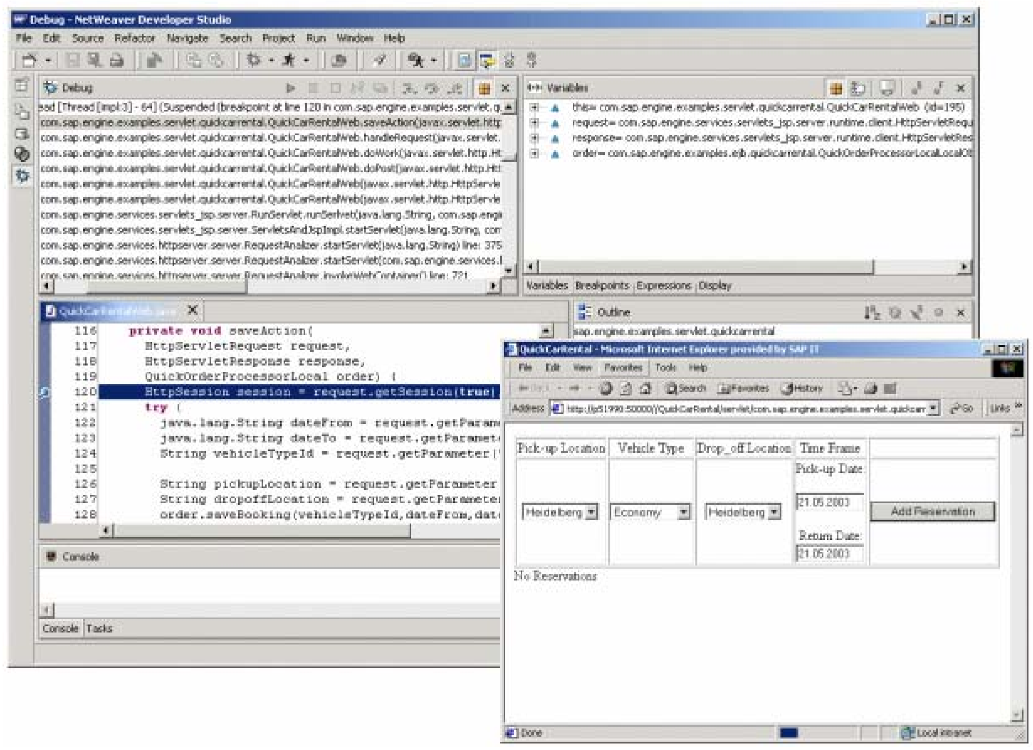
5. In the *Name* field, enter **`QuickCarRentalWeb`**.

   Choose the *Browse…* button next to the field *Enterprise Application Project*, select the project *J2EE_QuickCarRentalEar*. Choose *OK.*

   Choose the ReadioButton next to Servlet Click the *Choose…* button next to the field *Servlet:*, select the servlet *QuickReservationServlet*, and choose *OK*.

6. The configuration is now complete; you can start the debugger by choosing *Debug*.

7. The SAP NetWeaver Studio automatically switches to the debug perspective. The Web application is started in an external browser.
8. Enter valid date values in the input fields of the Web application and choose *Add Reservation*.
9. Change back to the SAP NetWeaver Developer Studio. The application was stopped at the breakpoint and can now be analyzed.


Debug - NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Variables
Variables Breakpoints Expressions Display
Outline
Console
Console Tasks
QuickCarRental - Microsoft Internet Explorer provided by SAP IT
File Edit View Favorites Tools Help
Search Favorites History
Address
Go
Links
Done
Local intranet

```
ead [Thread[impl:3] - 64] (Suspended (breakpoint at line 120 in com.sap.engine.examples.servlet.q
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.saveAction(javax.servlet.http
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.handleRequest(javax.servlet.
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.doWork(javax.servlet.http.Ht
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb.doPost(javax.servlet.http.Htt
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb(javax.servlet.http.HttpServle
com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb(javax.servlet.http.HttpServle
com.sap.engine.services.servlets_jsp.server.RunServlet.runServlet(java.lang.String, com.sap.engi
com.sap.engine.services.servlets_jsp.server.ServletsAndJspImpl.startServlet(java.lang.String, com
com.sap.engine.services.httpserver.server.RequestAnalizer.startServlet(java.lang.String) line: 375
com.sap.engine.services.httpserver.server.RequestAnalizer.startServlet(com.sap.engine.services.l
com.sap.engine.services.httpserver.server.RequestAnalizer.invokeWebContainer() line: 771
```

```
this= com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb (id=195)
request= com.sap.engine.services.servlets_jsp.server.runtime.client.HttpServletRequ
response= com.sap.engine.services.servlets_jsp.server.runtime.client.HttpServletRes
order= com.sap.engine.examples.ejb.quickcarrental.QuickOrderProcessorLocalLocalOt
```

QuickCarRentalWeb.java

```
    private void saveAction(
      HttpServletRequest request,
      HttpServletResponse response,
      QuickOrderProcessorLocal order) {
      HttpSession session = request.getSession(true);
      try {
        java.lang.String dateFrom = request.getParam
        java.lang.String dateTo = request.getParamete
        String vehicleTypeId = request.getParameter("

        String pickupLocation = request.getParameter
        String dropoffLocation = request.getParamete
        order.saveBooking(vehicleTypeId,dateFrom,dat
```

sap.engine.examples.servlet.quickcarrental

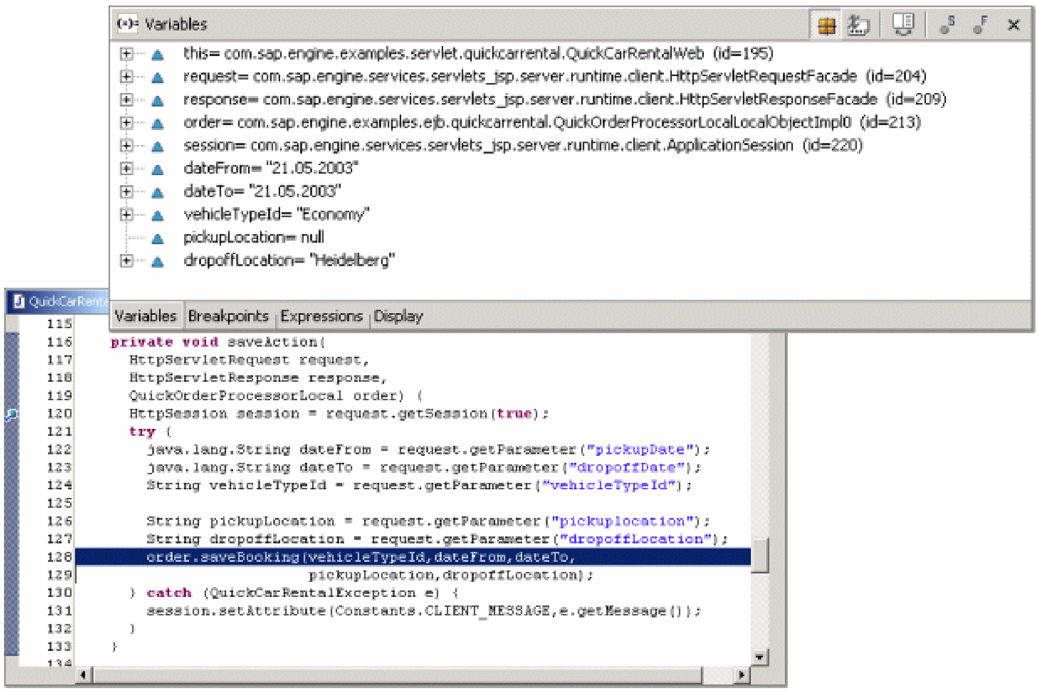
Pick-up Location	Vehicle Type	Drop_off Location	Time Frame	
			Pick-up Date:	
Heidelberg	Economy	Heidelberg	21.05.2003	Add Reservation
			Return Date:	
			21.05.2003	

No Reservations

You have the following debugging options:

Key	Description
F5: *Step Into*	Jumps to the next statement
F6: *Step Over*	The next command is executed without jumping to the current statement
F7: *Step Return*	If you previously chose F5, you can choose F7 to cancel the debugging of the current command

10. Use the `F6` key for executing all statements within the `saveObjects` method and observe the variable contents in the *Variables* window of the debug perspective.
11. The `pickupLocation` variable does not have a valid value (null) assigned to it.



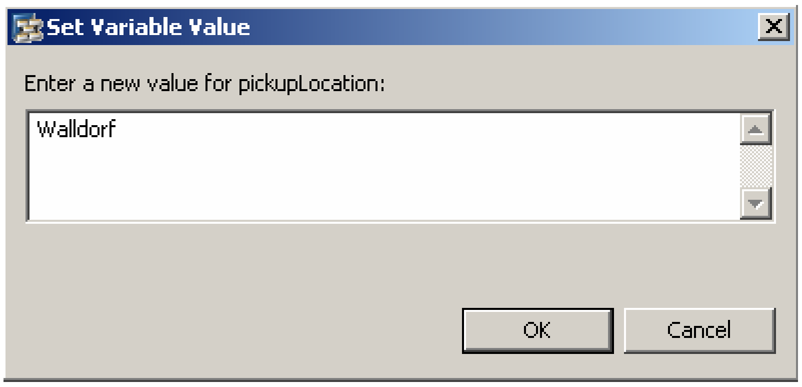
```
this= com.sap.engine.examples.servlet.quickcarrental.QuickCarRentalWeb  (id=195)
request= com.sap.engine.services.servlets_jsp.server.runtime.client.HttpServletRequestFacade  (id=204)
response= com.sap.engine.services.servlets_jsp.server.runtime.client.HttpServletResponseFacade  (id=209)
order= com.sap.engine.examples.ejb.quickcarrental.QuickOrderProcessorLocalLocalObjectImpl0  (id=213)
session= com.sap.engine.services.servlets_jsp.server.runtime.client.ApplicationSession  (id=220)
dateFrom= "21.05.2003"
dateTo= "21.05.2003"
vehicleTypeId= "Economy"
pickupLocation= null
dropoffLocation= "Heidelberg"
```

```
private void saveAction(
  HttpServletRequest request,
  HttpServletResponse response,
  QuickOrderProcessorLocal order) {
  HttpSession session = request.getSession(true);
  try {
    java.lang.String dateFrom = request.getParameter("pickupDate");
    java.lang.String dateTo = request.getParameter("dropoffDate");
    String vehicleTypeId = request.getParameter("vehicleTypeId");

    String pickupLocation = request.getParameter("pickuplocation");
    String dropoffLocation = request.getParameter("dropoffLocation");
    order.saveBooking(vehicleTypeId,dateFrom,dateTo,
                      pickupLocation,dropoffLocation);
  } catch (QuickCarRentalException e) {
    session.setAttribute(Constants.CLIENT_MESSAGE,e.getMessage());
  }
}
```

The debugging mode enables you to dynamically change the variable contents.

12. Double-click the `pickupLocation` variable in the *Variables* window of the debug perspective.
13. Enter a location in the displayed input field and proceed with debugging. The application will no longer generate error messages.



Enter a new value for pickupLocation:

Walldorf

Terminating Debugging

Exit debugging to correct the wrong assignment in the source code.

If you want to exit debugging, you must terminate the threads in the IDE (debug perspective).

1. In the *Debug* window, select the top node (*QuickCarRental[J2EE Application]*).
2. Right-click and choose *Terminate* from the context menu.

3. Then choose *Remove All Terminated* from the context menu.

4. Correct the source code of the servlet. Use the procedure described in Preparations for Debugging a Servlet [Seite 15] to restore the original status.