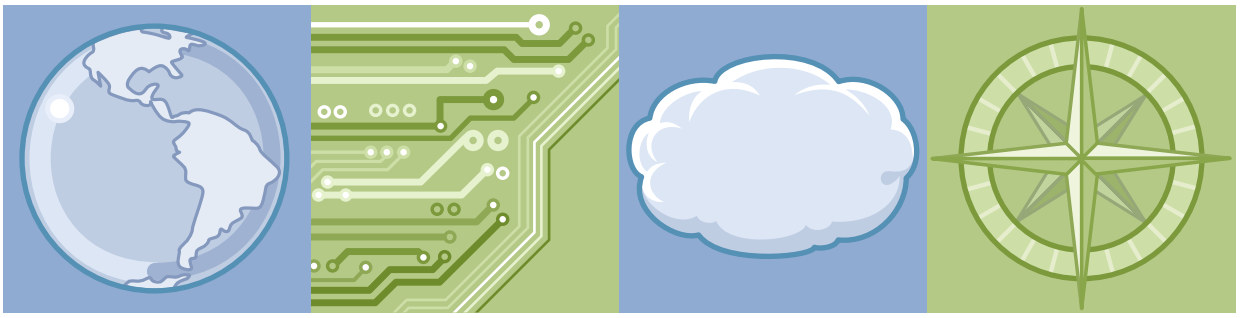
IBM Training

Student Exercises

WebSphere Application Server V8.5.5 Problem Determination

Course code WA591 ERC 3.0



IBM Cloud

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

AFS™	AIX 6™	AIX®
Approach®	DB™	DB2®
developerWorks®	HACMP™	IMS™
MVS™	PartnerWorld®	RACF®
Rational®	RDN®	Redbooks®
Redpaper™	Tivoli®	WebSphere®
WPM®	z/OS®	

Pentium is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

VMware and the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the "Marks") of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.

June 2017 edition

The information contained in this document has not been submitted to any formal IBM test and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer’s ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

© Copyright International Business Machines Corporation 2013.
This document may not be reproduced in whole or in part without the prior written permission of IBM.
US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Trademarks vii

Exercises description ix

Exercise 1. Using the IBM Support Assistant Team Server 5.0 1-1
Section 1: Starting the IBM Support Assistant 1-3
Section 2: Log in to the IBM Support Assistant Administration Console and explore it . . 1-13
Section 3: Create a case and launch tools from the Tools tab 1-16
Section 4: Optional: Scan a case 1-26
Section 5: Run the IBM Support Assistant Data Collector tool 1-33

Exercise 2. Gathering diagnostic data 2-1
Section 1: Working with log files of the application server 2-3
Section 2: Enable and configure HPEL 2-9
Section 3: Use the Log Viewer in the administrative console to examine log data and trace data 2-15
Section 4: Enable tracing for an application server and view trace data by using the Log Viewer 2-17
Section 5: Enable cross-component trace (XCT) 2-21
Section 6: Examine runtime messages 2-26
Section 7: Examine FFDC logs 2-27
Section 8: Work with the Classloader viewer 2-28
Section 9: Work with namespace tools 2-31

Exercise 3. Introduction to configuring garbage collection policies 3-1
Section 1: Enable verbose GC for an application server 3-3
Section 2: Change the GC policy for an application server 3-7
Section 3: Use IBM Monitoring and Diagnostic Tools for Java – Garbage Collection and Memory Visualizer (GCMV) to analyze verbose GC data 3-11
Section 4: Configure and use the Java Health Center 3-15

Exercise 4. Troubleshooting hung threads 4-1
Section 1: Increase the maximum heap size for the deployment manager 4-2
Section 2: Install the test application 4-3
Section 3: Manually initiate a javacore dump 4-4
Section 4: Configure the hang detection policy 4-7
Section 5: Generate and analyze a deadlock scenario 4-10
Section 6: Generate and investigate hung application threads 4-15

Exercise 5. Troubleshooting crashes 5-1
Section 1: Start the servers 5-2
Section 2: Analyze a javacore to troubleshoot a server crash 5-2
Section 3: Use the Memory Analyzer to examine a system core file 5-5
Section 4: Use the Interactive Diagnostic Data Explorer (IDDE) to analyze system core files 5-14

Exercise 6. Troubleshooting an out-of-memory condition 6-1

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

6.1. Explore and configure tools for gathering JVM heap diagnostic data 6-2
Section 1: Start all servers and enable verbose garbage collection for server1 6-2
Section 2: Use Tivoli Performance Viewer to monitor the JVM of server1 6-4
Section 3: Trigger a heap dump for server1 and use the Heap Analyzer tool to examine it 6-7
6.2. Troubleshoot an application that has a Java out-of-memory error 6-11
Section 1: Run the BadApp application and generate the OutOfMemoryError exception 6-11
Section 2: Use tools to analyze OutOfMemoryError exception artifacts 6-17
Section 3: Use the Memory Analyzer to view the system core 6-29

Exercise 7. Troubleshooting database connection problems 7-1
7.1. Simulate a database connection problem and collect diagnostic data. 7-2
Section 1: Preparing the environment 7-2
Section 2: Investigate the symptoms of the misconfigured data source 7-3
7.2. Troubleshoot problems when connecting to a remote database server 7-12
7.3. Troubleshoot a database connection problem 7-15

Exercise 8. Troubleshooting a connection leak 8-1
Section 1: Preparing the environment 8-2
Section 2: Load testing the application 8-5
Section 3: Determining the cause of the connection leak 8-10
Section 4: Resolving the connection leak 8-15
Section 5: Testing the updated application 8-18
Section 6: Clean up the environment 8-19

Exercise 9. Troubleshooting security problems 9-1
Section 1: In this section, you start the IBM Tivoli Directory Server on your lab image and verify that the server can be accessed. 9-2
Section 2: Verify the startup of the deployment manager and node agent 9-4
Section 3: Change security realm for administrative and application security 9-6
Section 4: Simulate an inaccessible LDAP server and troubleshoot the problem 9-12
Section 5: Troubleshooting authentication failures 9-17
Section 6: Troubleshooting an authorization failure 9-19
Section 7: Tracing security components 9-24
Section 8: Re-enable the file-based federated repository 9-26
Section 9: Troubleshooting an SSL handshake failure 9-30
Section 10: Disable administrative security 9-42

Exercise 10. Troubleshooting server start failures 10-1
Section 1: Prepare your lab image environment 10-2
Section 2: Simulate and troubleshoot an application server start failure 10-3
Section 3: Simulate and troubleshoot a node agent start failure 10-5
Section 4: Simulate and troubleshoot a deployment manager start failure 10-9

Exercise 11. Troubleshooting request flow and web container problems 11-1
Section 1: Prepare the environment and change log levels 11-2
Section 2: Examine the results of a successful web request 11-7
Section 3: Examine the results of a failed web request 11-9
Section 4: Examine failures at the plug-in 11-10
Section 5: Examine a failed WebSphere Application Server request 11-14
Section 6: Configuring a diagnostic provider on the web container 11-20

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 7: Use IBM Web Server Plug-in Analyzer for WebSphere Application Server [Desktop] to analyze plug-in data 11-25

Exercise 12. Troubleshooting WebSphere default messaging 12-1
Section 1: Start the deployment manager and create a second application server 12-2
Section 2: Create a bus and add two application servers to the bus as members 12-3
Section 3: Create a bus destination 12-8
Section 4: Connect to the bus with the IBM Service Integration Bus Destination Handler 12-11
Section 5: Produce a message by using the IBM Service Integration Bus Destination Handler 12-13
Section 6: Did the message flow as expected? 12-15
Section 7: Consume the message by using the IBM Service Integration Bus Destination Handler 12-17
Section 8: Explore message flow when an ME hosting a destination is down 12-20

Exercise 13. Configuring health management policies 13-1
Section 1: Install the BadAppEARProject application if required 13-3
Section 2: Configure settings for exercise environment 13-3
Section 3: Configure a health policy 13-7
Section 4: Test Health Management 13-18

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

AFS™	AIX 6™	AIX®
Approach®	DB™	DB2®
developerWorks®	HACMP™	IMS™
MVS™	PartnerWorld®	RACF®
Rational®	RDN®	Redbooks®
Redpaper™	Tivoli®	WebSphere®
WPM®	z/OS®	

Pentium is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

VMware and the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the "Marks") of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercises description

This course includes the following exercises:

- Exercise 1: Using the IBM Support Assistant
- Exercise 2: Introduction to problem determination tools
- Exercise 3: Introduction to configuring garbage collection policies
- Exercise 4: How to troubleshoot hangs
- Exercise 5: How to troubleshoot crashes
- Exercise 6: Troubleshooting out-of-memory condition
- Exercise 7: Troubleshooting database connection problems
- Exercise 8: Troubleshooting a connection leak
- Exercise 9: Troubleshooting security problems
- Exercise 10: Troubleshooting server start failures
- Exercise 11: Troubleshooting request flow and web container
- Exercise 12: Troubleshooting WebSphere default messaging
- Exercise 13: Health Management

In the exercise instructions, you can check off the line before each step as you complete it to track your progress.

Most exercises include required sections that should always be completed. It might be necessary to complete these sections before you can start later exercises. Some exercises also include optional sections that you might want to complete if you have sufficient time and want an extra challenge.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 1. Using the IBM Support Assistant Team Server 5.0

What this exercise is about

This exercise demonstrates how to use the IBM Support Assistant.

What you should be able to do

At the end of this exercise, you should be able to:

- Start and stop the IBM Support Assistant Team Server
- Administer the IBM Support Assistant Team Server
- Run report generator tools and examine the reports
- Use the Case Manager to create a case and add diagnostic data
- Run interactive desktop tools
- Use Automated Analysis to scan a case and examine the results
- Run the ISA Data Collector tool

Introduction

IBM Support Assistant is a web-based application to assist with organizing, analyzing, and diagnosing issues with software. It applies the concept of cases to grouping problem diagnostic files together regarding an overlying issue. Using an intuitive interface, it provides easy case management, file management, problem determination capabilities, and automated data collection capabilities.

IBM Support Assistant 5 includes its own server that is easily configured and maintained. IBM Support Assistant can be installed as a Team Server so that a group of users can all access a single, shared instance of IBM Support Assistant by pointing their browsers to it. Alternatively, it can also be installed as a single-user desktop application and can be accessed through a browser by pointing to the local server.

Requirements

All of the files necessary for this exercise are stored in a special directory on your lab computer. Some of the activities require access to the Internet, but these activities are optional.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

IBM Support Assistant Team Server 5 must be installed.

Important

New versions of IBM Support Assistant tools

A new course image was released in February 2016. It has an updated version of IBM Support Assistant Team Server 5.0.2.1 and updated tools with valid security certificates.

You might notice slight variations between some of the screen captures in the exercises and how the new versions of the tools appear. In general, the functions that are required for these exercises are unaffected in this edition of the course material.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles

Section 1: Starting the IBM Support Assistant

The IBM Support Assistant Team Server is an application server that hosts Java EE applications. In particular, the `ISA.ear` application is installed which implements the web client and other services. All of the problem determination tools are installed to the server as WAR files. The Memory Analyzer web edition is deployed as a WAR file and can run on a dedicated application server.

__ 1. Start the IBM Support Assistant from the command line.

__ a. From a Terminal window, change directory to: `/opt/IBM/ISA/ISA5`

__ b. Enter the command: `./start_isa.sh`

```
root@washost:/opt/IBM/ISA/ISA5
File  Edit  View  Search  Terminal  Help
[root@washost ISA5]# ./start_isa.sh

Now starting IBM Support Assistant Team Server.
System resources and system load may affect the time required
to start the application.  Please be patient...

Starting server isa.
Server isa started with process ID 6699.
---------------------------------------------------------------------------

Starting server com.ibm.java.web.memoryanalyzer.
Server com.ibm.java.web.memoryanalyzer started with process ID 6785.
---------------------------------------------------------------------------

Starting server com.ibm.java.web.idde.
Server com.ibm.java.web.idde started with process ID 6844.
---------------------------------------------------------------------------

IBM Support Assistant is ready to run.
Open a browser to:
http://localhost:10911/isa5   OR   http://<hostname>:10911/isa5
---------------------------------------------------------------------------
Press ENTER to finish...
```

__ c. Do ***not*** press Enter. Minimize this window.

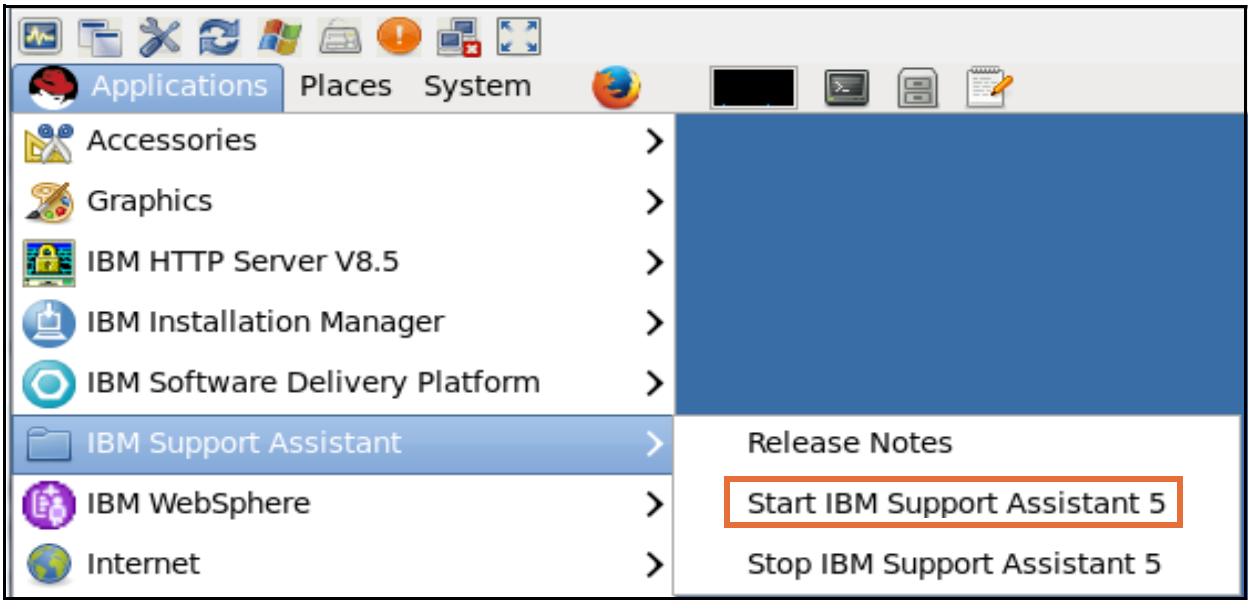
__ d. Notice that this command starts three servers, the IBM Support Assistant server, a server for the Memory Analyzer, and a server for the Interactive Diagnostic Data Explorer (IDDE) web-based tool. For the IBM Support Assistant deployment on your lab image, each of these servers is a WebSphere Liberty Profile server.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

Start and stop shortcut

You can also start and stop the IBM Support Assistant from the Linux desktop by clicking **Applications** and selecting **IBM Support Assistant > Start IBM Support Assistant 5**.



__ 2. Start the IBM Support Assistant client in a web browser.

__ a. Start a Firefox web browser and enter the web address:

`http://localhost:10911/isa5`

Note

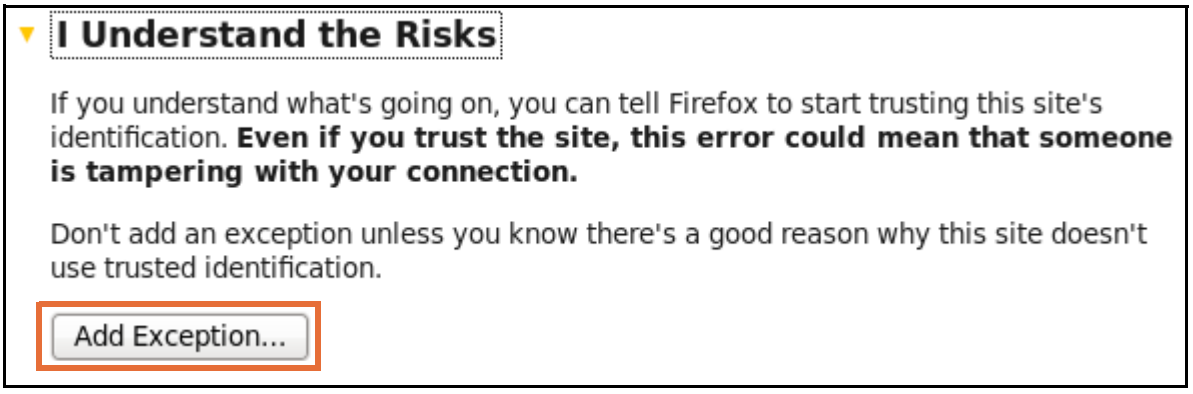
If you do not see the message "**This Connection is Untrusted**", skip to step e.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. The browser shows a message that the server connection is untrusted. The screen shows how Firefox shows this message: Click "**I understand the Risks**".

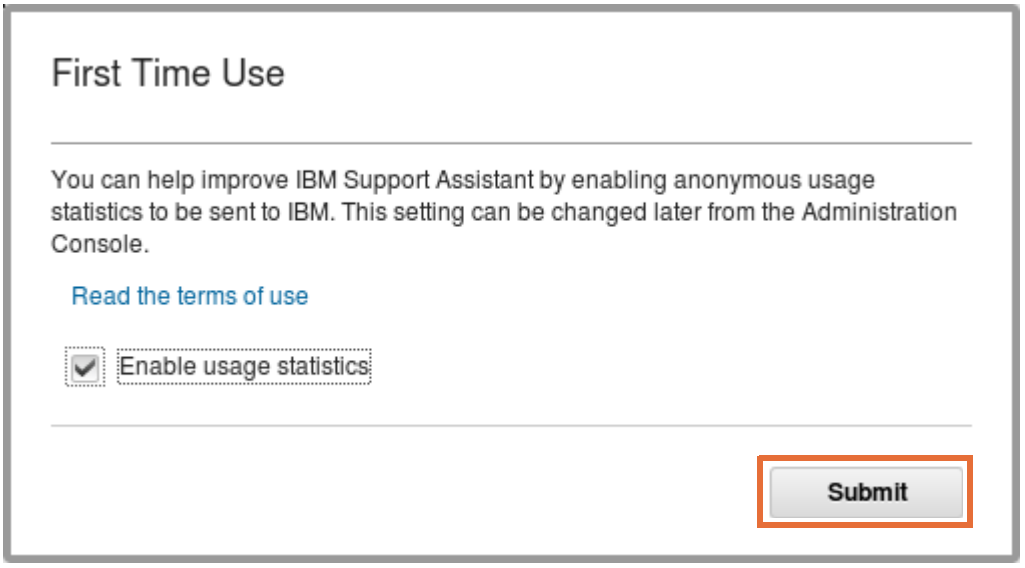
__ c. Firefox shows this screen. Click **Add Exception...**



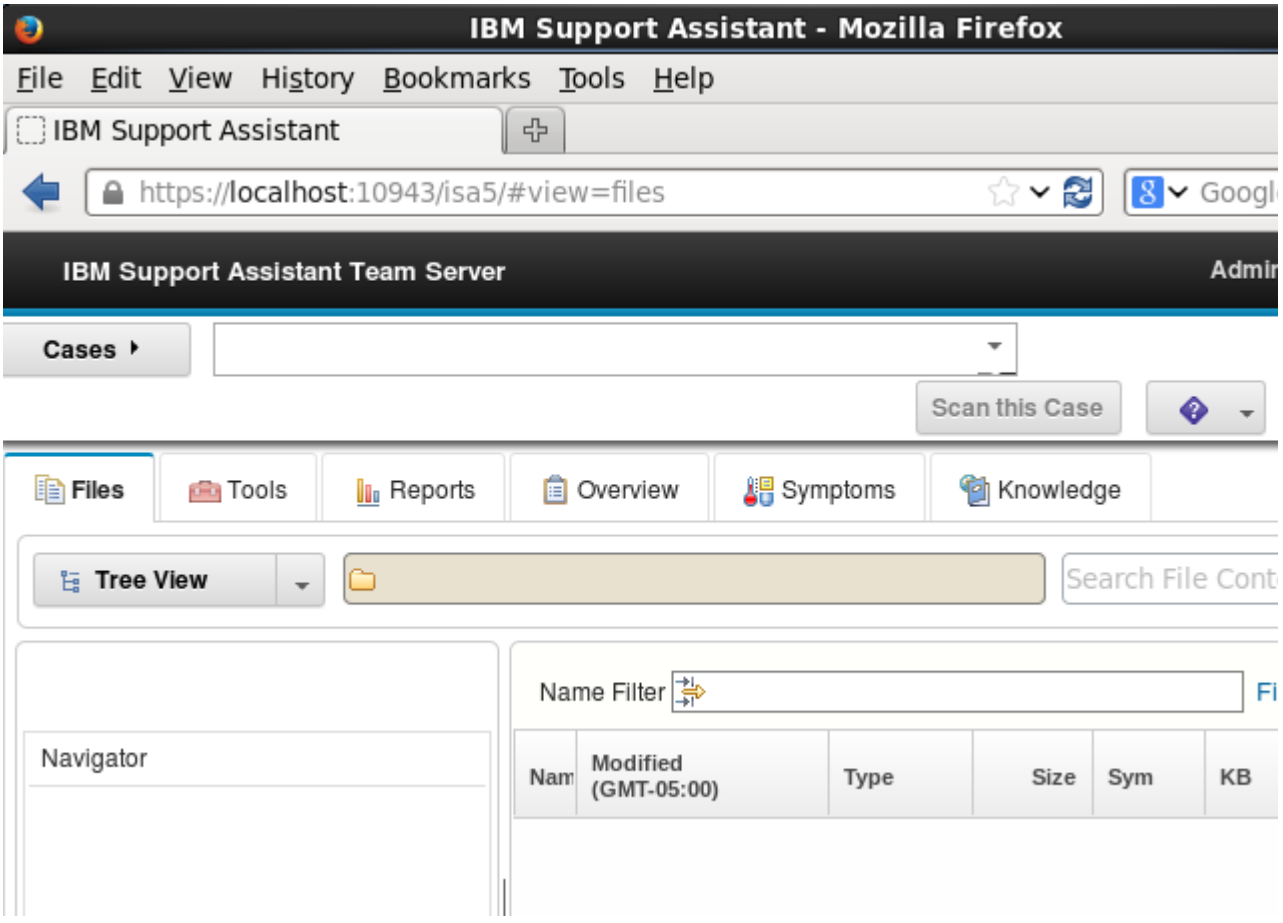
__ d. The next screen provides detail about this security exception. Examine this screen; then click **Confirm Security Exception**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. The first time that you start IBM Support Assistant you receive a request to send usage statistics to IBM. Click **Submit**.



__ f. The IBM Support Assistant home page opens in the browser.



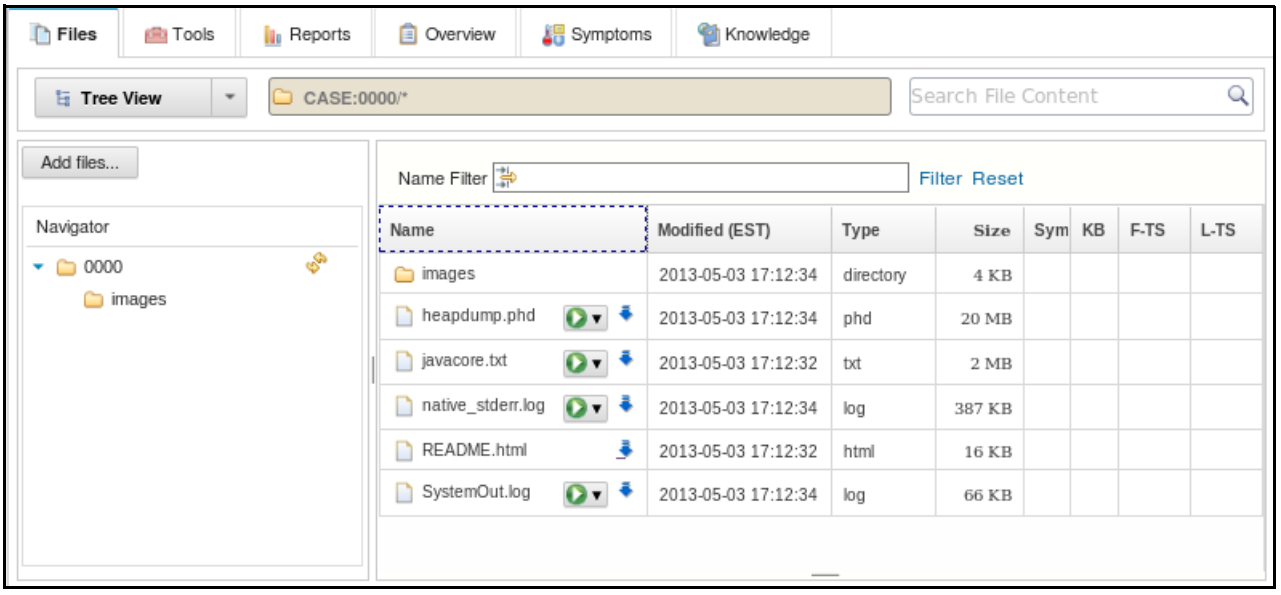
__ g. Bookmark IBM Support Assistant in the browser for future use.

© Copyright IBM Corp. 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 3. Explore some of the features of the IBM Support Assistant web page.

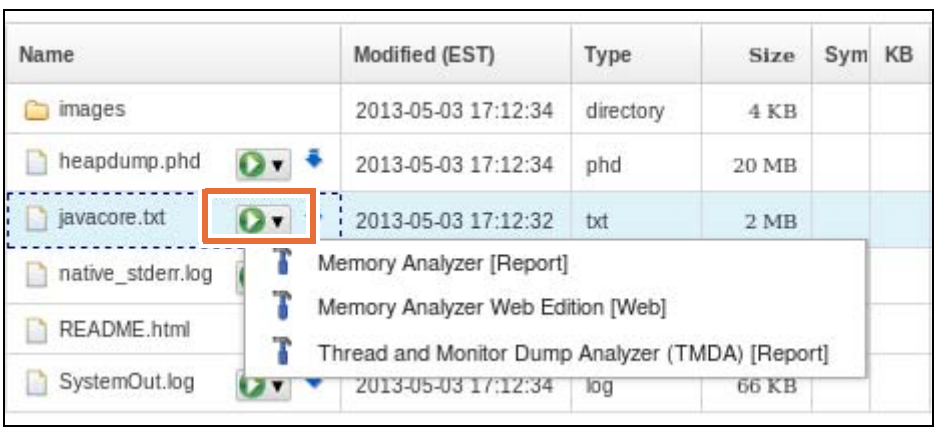
__ a. Use the menu beside the Cases button to select **[0000] Example Case**.

__ b. When the case is selected, the Files tab comes into focus and the Navigator pane displays folders that contain diagnostic data for this case.



__ c. Notice that the right pane shows the files and folder that are contained in the folder for Case 0000. In this example case, diagnostic data was added in the form of `heapdump.phd`, `javacore.txt`, `native_stderr.log`, and `SystemOut.log` files.

__ d. Each of these files has a green Quick Tool Launcher beside its name. Click the launcher on the **javacore.txt** file to the preferred tools for processing this file.



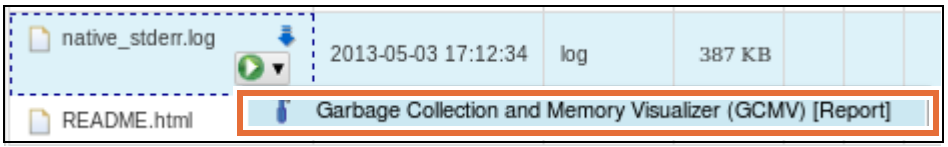
__ e. It is also possible to launch a tool that is not listed as a preferred tool. Right-click the name of a file and from its menu, select **Run Tool > Other** to see a list of all tools.

__ f. Examine each of the other files in this case to see what tools are available for it. Selecting any of the tools that are listed launches the tool against the diagnostic data file.

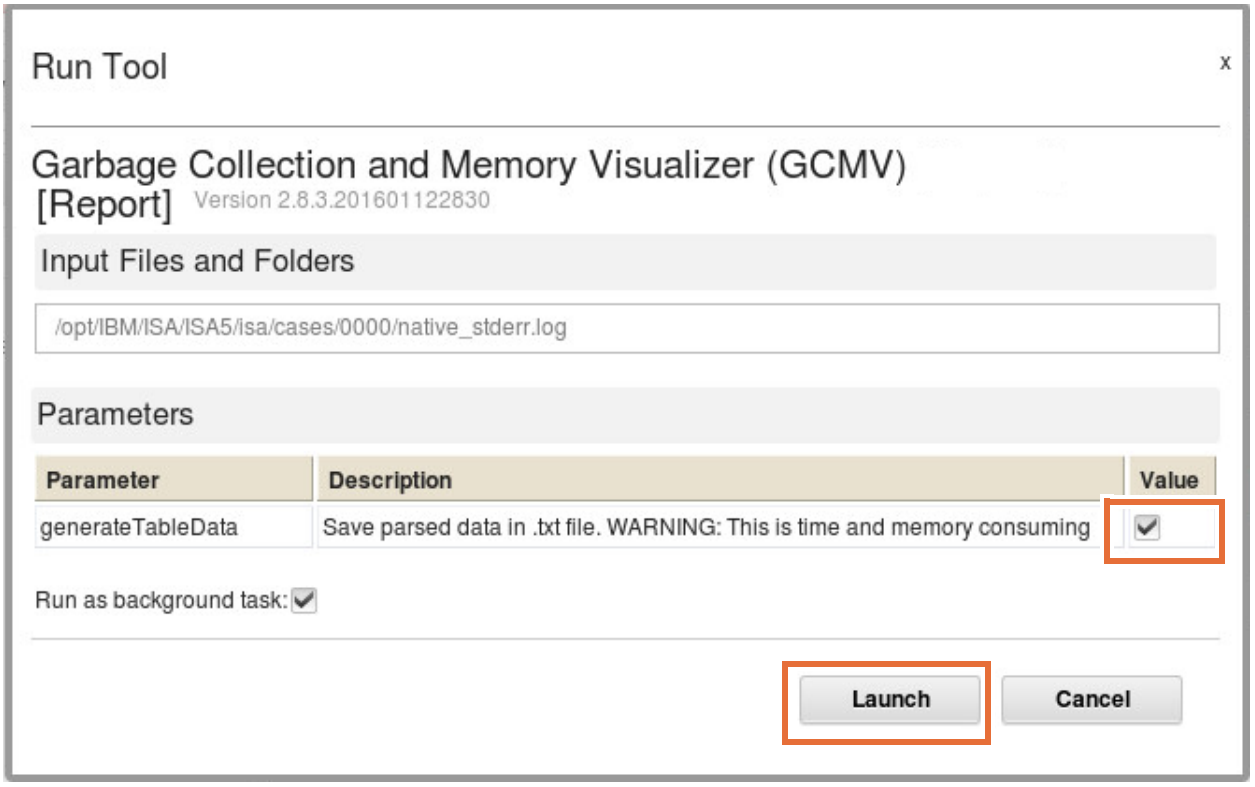
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. Use the Quick Tool Launcher to run Garbage Collection and Memory Visualizer (GCMC) against the native_stderr.log file.

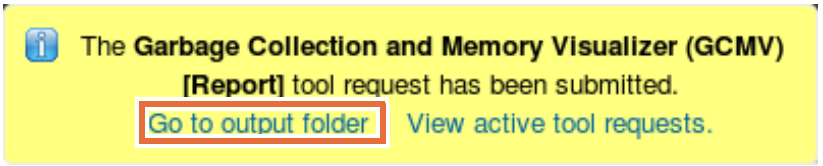
__ a. Click the launcher for the **native_stderr.log** file and select **GCMV [Report]** on the menu.



__ b. Select the **generateTableData** check box, and click **Launch**.



__ c. On the message, click the link **Go to output folder**.



Note

Run as background task check box

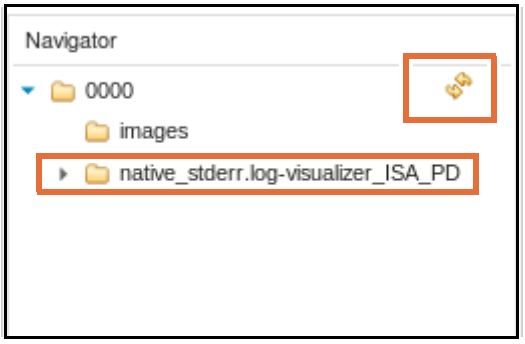
This check box is selected by default. This option is an artifact, and the behavior for running the tool is the same regardless of whether the check box is selected or not. When the check box is selected, you immediately see a yellow alert message with the link “Go to output folder.” If the check box is

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

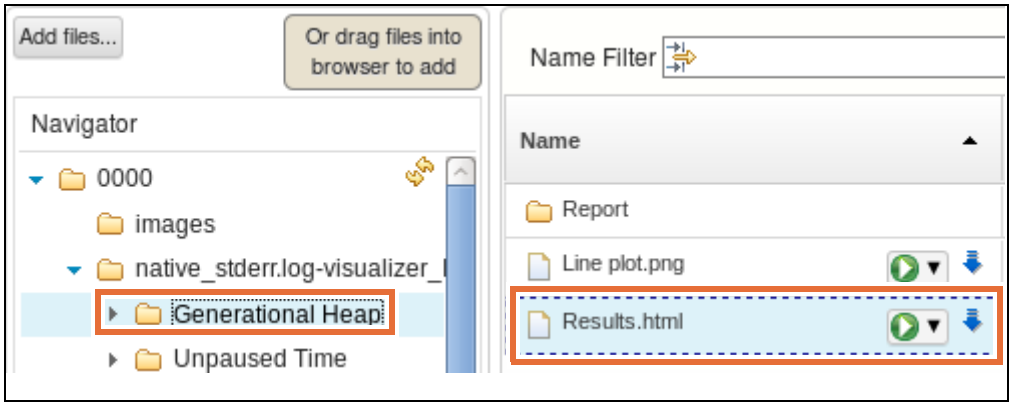
not selected, then you see a yellow alert message when the tool finishes running and you see a link to the report.

__ d. The GCMV tool can take several minutes to analyze the `native_stderr.log` file and generate a report. The report shows as a new folder in the Navigator pane.



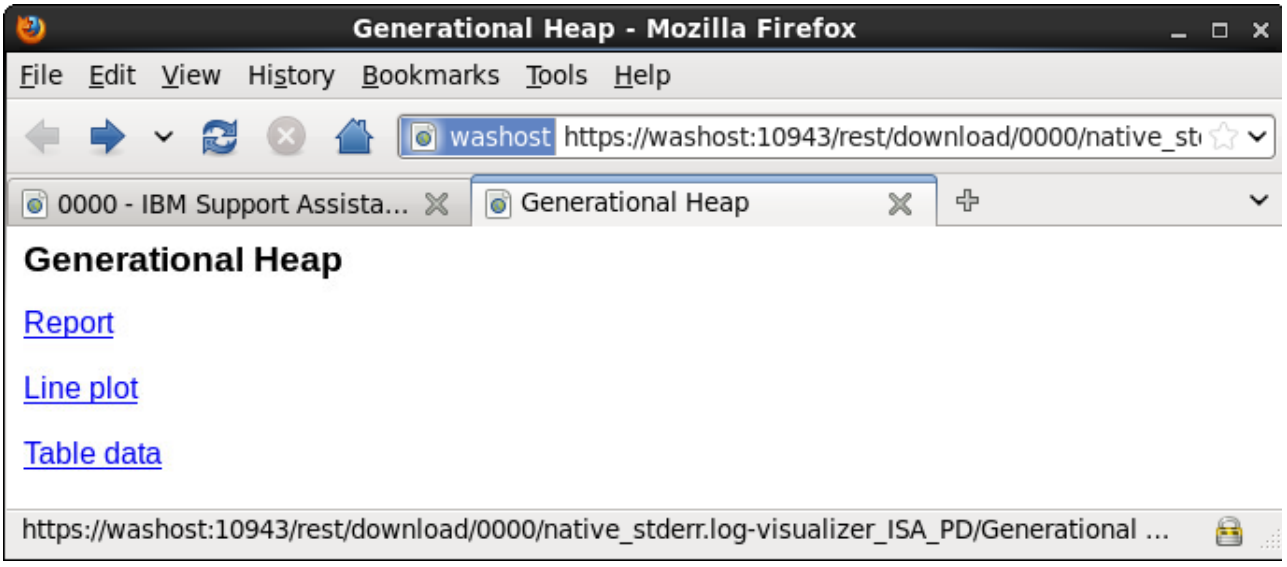
__ e. Wait until the report is available. Click the refresh icon in the Navigator. Select or expand the **native_stderr.log-visualizer_ISA_PD** folder to view its contents.

__ f. Select the **Generational Heap** folder and double-click the **Results.html** file in the right pane.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. The results are displayed in a new browser tab.

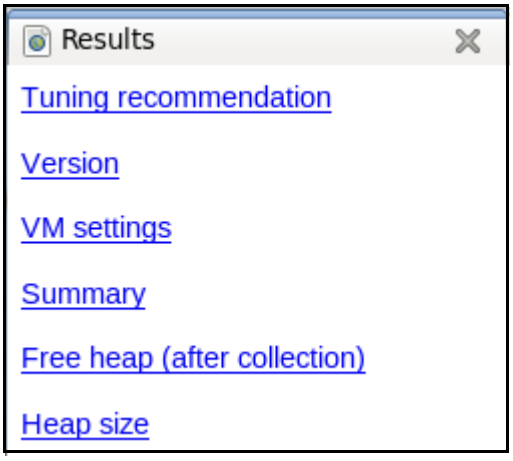


__ h. Click the **Report** link. Examine the contents of the report.

Information

Reviewing the analysis report

GCMV generates a simple high-level report with links to the full report, a line plot of the data, and a table of the data. The full report shows you a table of contents on the left and a series of analysis sections.



For this exercise, you want to focus on the **Tuning Recommendations** section. You notice an important alert and several warnings as indicated by the red **X** icon and yellow triangles.

As you read through the tuning recommendations, you see that GCMV alerts you that the application appears to be leaking memory. More warnings about the use of large objects and long

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

garbage collection pause times are provided. It might be that you missed something in your coding and testing, but at least GCMV alerts you to a problem.

Tuning recommendation

Your application appears to be leaking memory. This is indicated by the used heap increasing at a greater rate than the application workload (measured by the amount of data freed). To investigate further see Diagnostics Guide.

At one point 30336 objects were queued for finalization. Using finalizers is not recommended as it can slow garbage collection and cause wasted space in the heap. Consider reviewing your application for occurrences of the finalize() method. You can use the ISA Tool Add-on, IBM Monitoring and Diagnostic Tools for Java - Memory Analyzer to list objects that are only retained through finalizers.

Your application is allocating many large objects, which affects performance.Consider increasing -Xloaminimum and -Xloainitial.

Garbage collection is causing some large pauses. The largest pause was 12215 ms. This may affect application responsiveness. If responsiveness is a concern then a switch of policy or reduction in heap size may be helpful.

Scroll down through the report to see data about VM settings, Free heap, and Heap size.

__ 5. View the Line plot.

__ a. Use the back arrow on your browser, and click the **Line plot** link to download a graph of the plotted data from the `native_stderr.log` file.

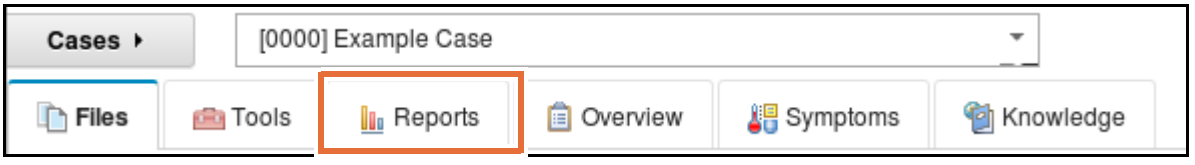
__ b. The line plot shows Heap size (top line) and Free heap after collection (bottom line).

__ c. In this exercise, you are just exploring the capabilities of the IBM Support Assistant. In later lectures and exercises, you learn more about interpreting the diagnostic data.

__ d. Close the Generational Heap browser tab by clicking the **X**.

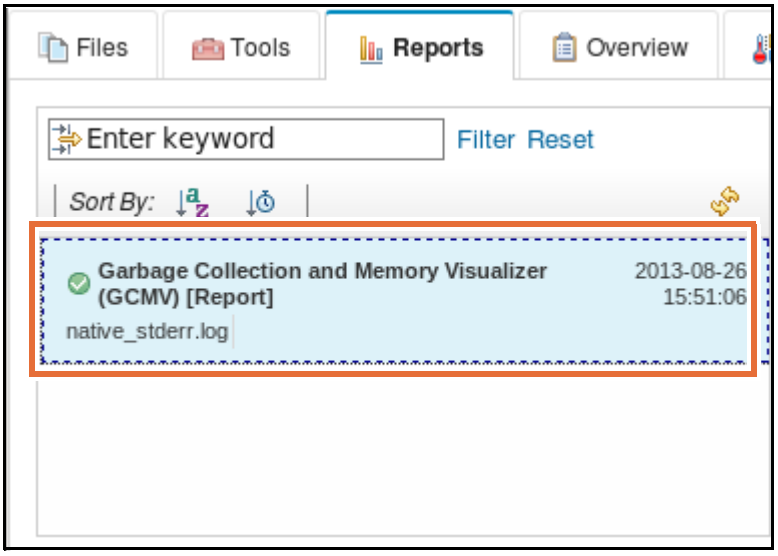
__ 6. It might be easier to use Reports tab to view reports. Reports from the report version of a tool are also saved in the Reports tab for a case.

__ a. Click the **Reports** tab.



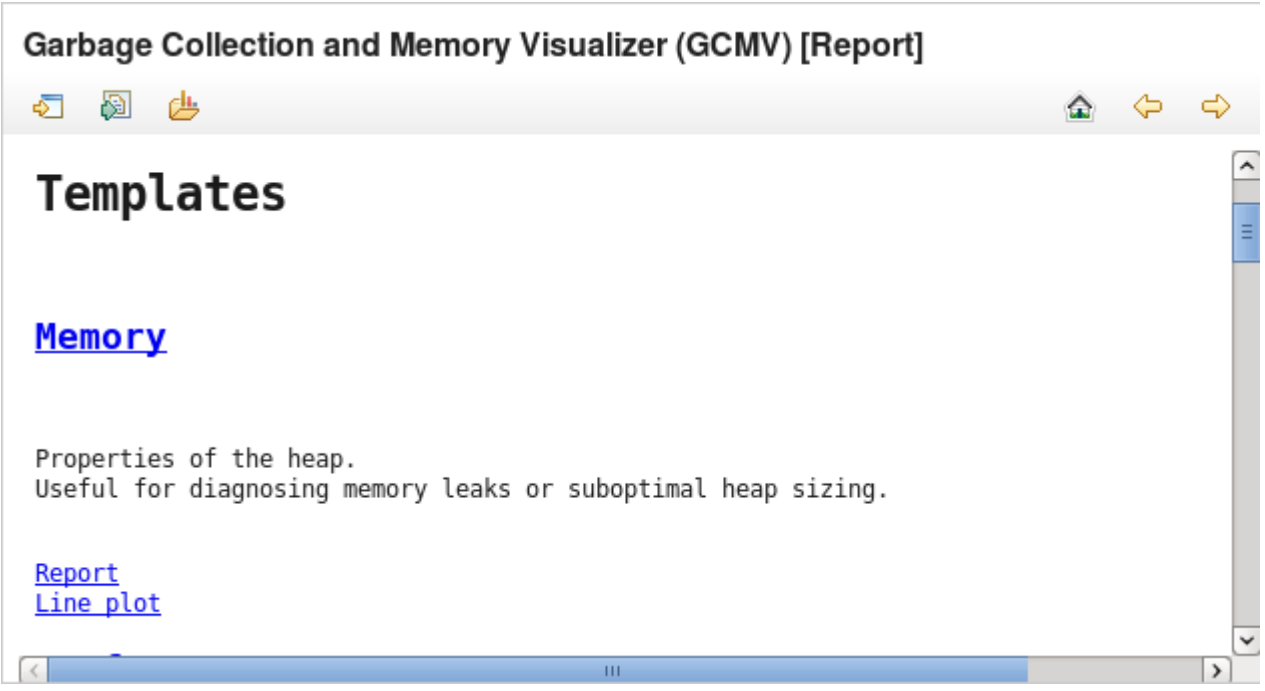
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. All reports that were generated for the case are listed in the left pane.



__ c. If you do not see any reports, click the Refresh icon.

__ d. Click the report to view its contents.



© Copyright IBM Corp. 2013

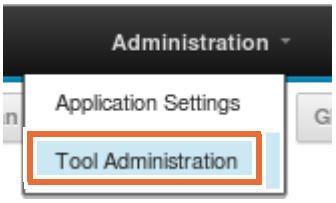
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 2: Log in to the IBM Support Assistant Administration Console and explore it

The Tools Administration tab in the administration console shows a list of all problem determination tools that are available for IBM Support Assistant 5. An administrator can log in to this tab and install, update, and uninstall problem determination tools.

__ 1. Log in to the IBM Support Assistant administration console.

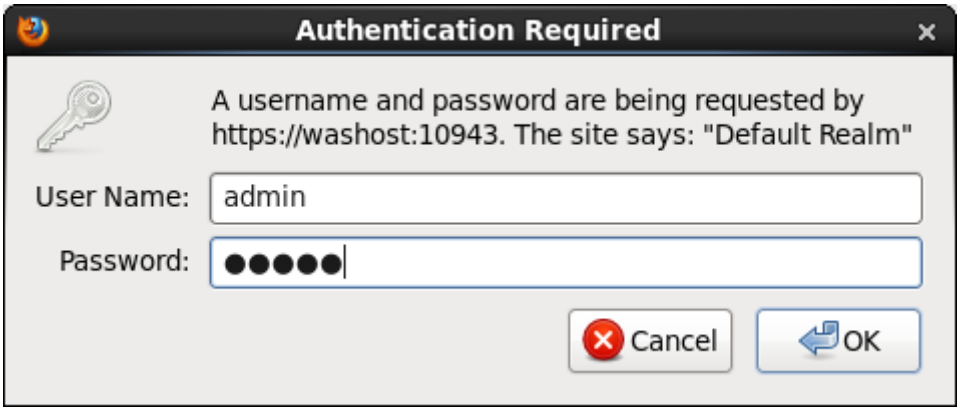
__ a. On the banner for IBM Support Assistant, click the **Administration** menu and select **Tools Administration**.



__ b. When you are prompted to authenticate, enter:

User Name: `admin`

Password: `admin`

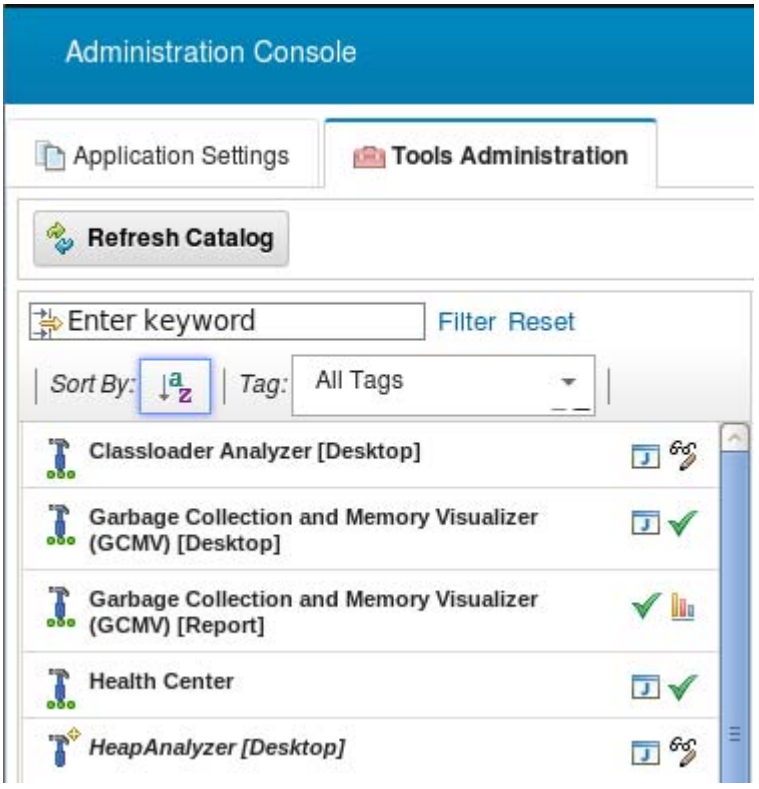


__ c. You are logging in with the default IBM Support Assistant administrator credentials. They can be changed by editing the `server.xml` file for the IBM Support Assistant server. Click **OK**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Explore the Tools Administration console.

__ a. After logging in, you see the catalog of IBM Support Assistant tools or the “Tool catalog.”



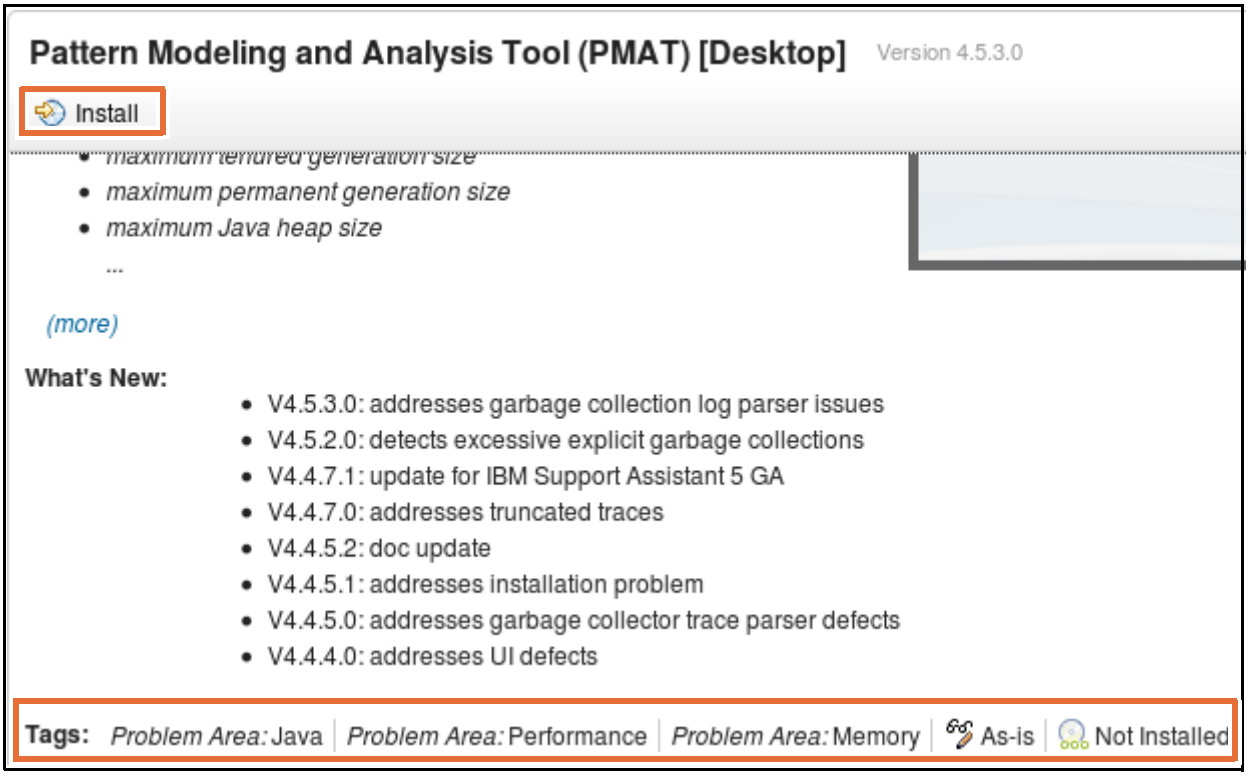
__ b. Try clicking **Refresh Catalog** and sorting the catalog. Try displaying different tags such as **Installed** and **Not Installed**; click **Refresh Catalog**.

__ c. Recall from the lecture that the catalog uses visual clues to signal the status of a tool: Installed, Not Installed, Update Available. You can also hover the cursor over any tool in the catalog to see information about its status. Try hovering the cursor over any of the tools.

© Copyright IBM Corp. 2013

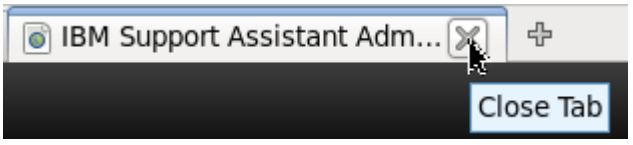
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Click one of the tools in the catalog that are not installed, such as **PMAT [Desktop]**. Click its name to see details about it in the right pane.



__ e. Notice that you can see a description of the tool, what is new, and the tags for the tool. Since the tool is not installed, you can use an Install link to install the tool. To install a tool, you need Internet access and an IBM ID. Installation can take a while to complete, so you are not going to install tools in this lab exercise.

__ 3. Continue to explore other tools by clicking their names in the catalog, as time permits. When you are done exploring, close the IBM Support Assistant Administration Console tab.



Note

IBM Support Assistant administrator credentials

The administrator user name and password are cached in the browser. So you must clear the browser cache; otherwise, a user is not prompted to log in, and the administration console is not protected.

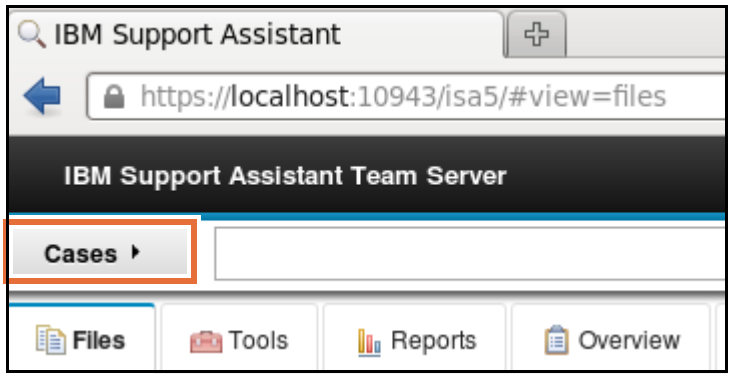
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 3: Create a case and launch tools from the Tools tab

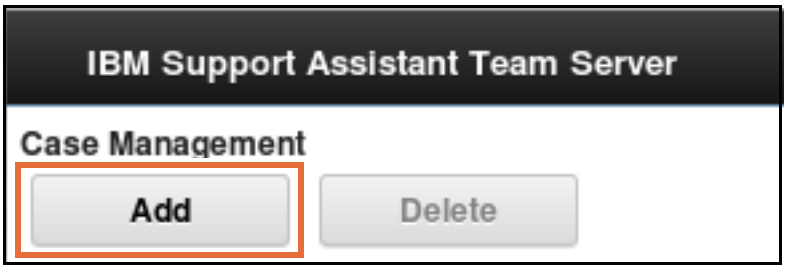
Before you begin troubleshooting a problem, you can create a case to manage related diagnostic files. Cases help you to organize diagnostic artifacts by problem incident, failing system, time, or any other logical categories that you like. When you create a case, you assign a summary and you can optionally add details.

__ 1. Create a case.

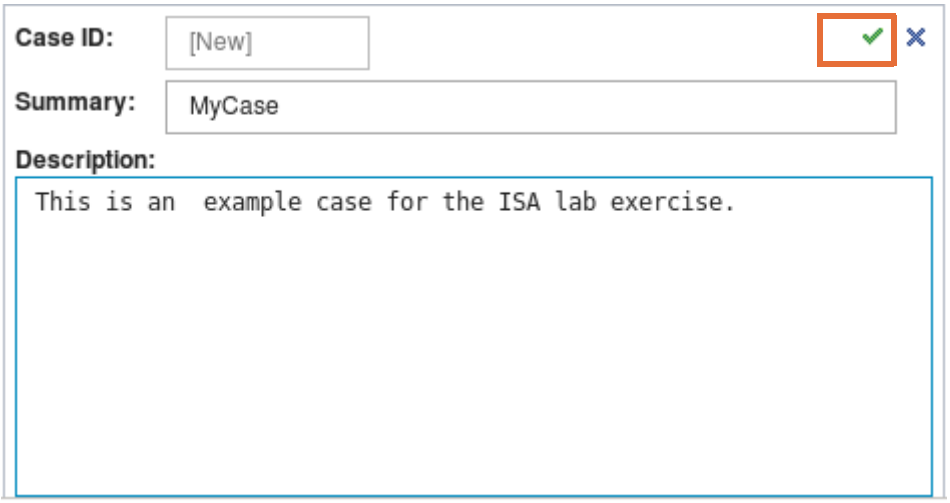
__ a. On the IBM Support Assistant home page, click **Cases**.



__ b. In the Case Management dialog box, click **Add**.



__ c. Enter **MyCase** for the Summary, and provide an optional description.



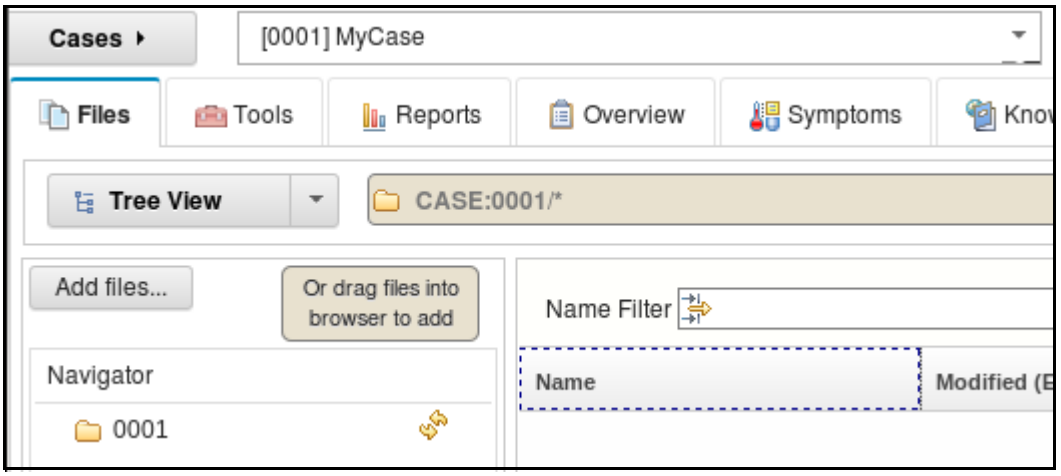
__ d. Notice that the Case ID field is not editable. The Case ID is automatically assigned when you save the case; click the **green check mark** to save the case.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

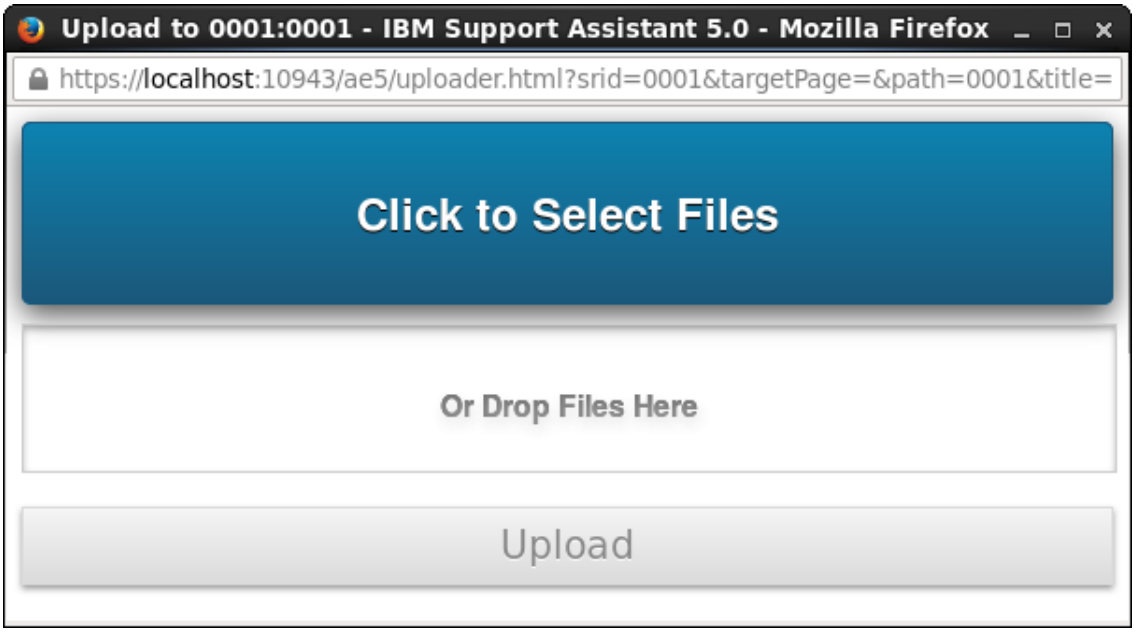
__ e. Notice that the Case ID that is assigned is 0001. Click anywhere outside of the Case Management dialog box to exit it.

__ f. Make sure that **[0001] MyCase** shows up in the case window, and that the **Files** tab is selected.



__ 2. Add diagnostic data files to the case. In the rest of this course, you learn how to gather diagnostic data. For this exercise, a set of diagnostic data files is provided for you.

__ a. Click **Add files.** A window comes up to select files and upload them.



__ b. You can also add files by dragging them into the browser. If your browser supports dragging, you see **"Or drag files into the browser to add"** beside the Add Files button.

__ c. Click **"Click to Select Files"** and navigate to: `/usr/Student/ISA_Lab/MyCase_Files`

__ d. Select the file **native_stderr.log** and click **Open**.

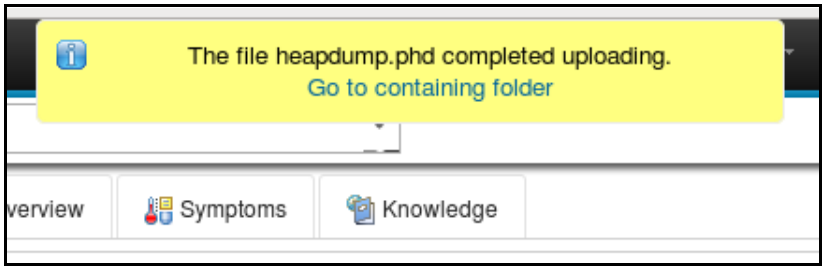
__ e. When you see the file in the window, click **Upload**.

__ f. When the upload completes, click **Select More Files**, and upload the other two files.

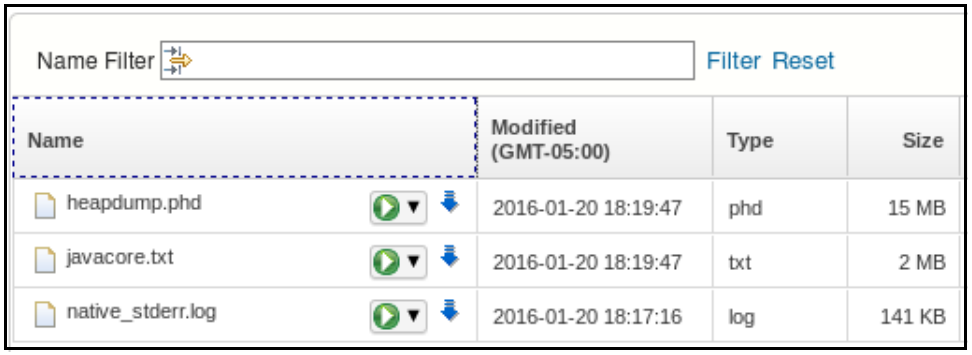
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. Close the upload window.

__ h. You see a message that shows the file completed uploading.



__ i. Click the **"Go to containing folder"** link to see the files that you uploaded.



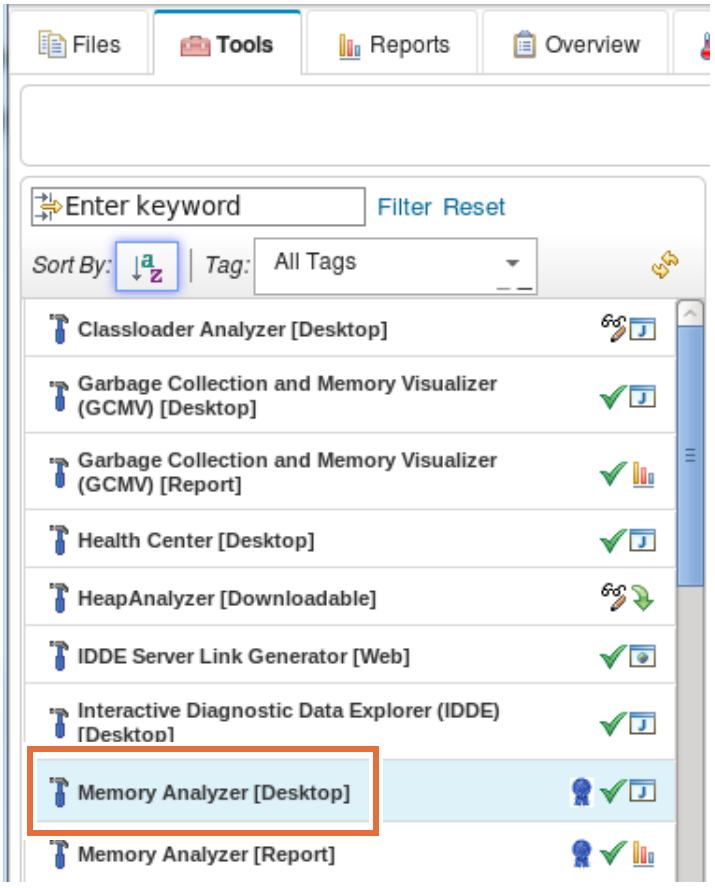
__ 3. Launch tools from the Tools tab. Earlier in this lab exercise you learned how to use the Quick Tool Launcher to run a report tool, but desktop tools must be launched from the Tools tab.

__ a. From the IBM Support Assistant, select the **Tools** tab.

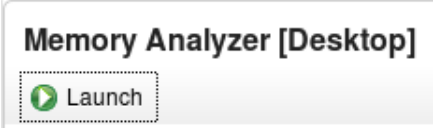
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Scroll down the list of tools and click **Memory Analyzer [Desktop]**.

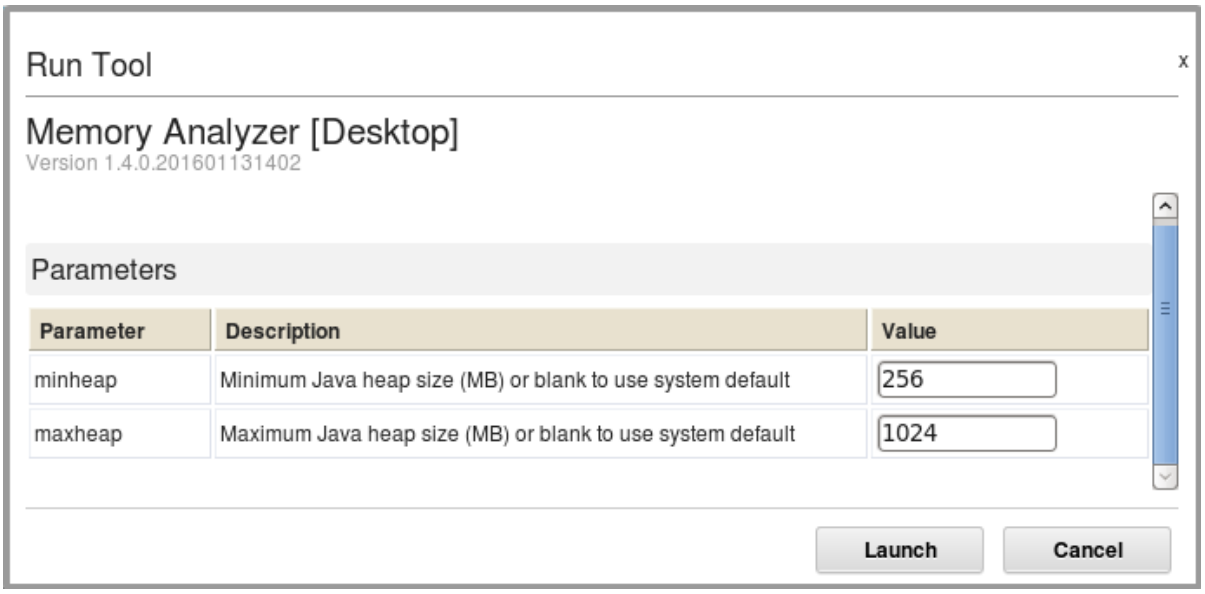


__ c. On the tool description pane, you see a Launch button. Notice that this view is different from the Administration console view where you would see an Uninstall link. Click **Launch**.

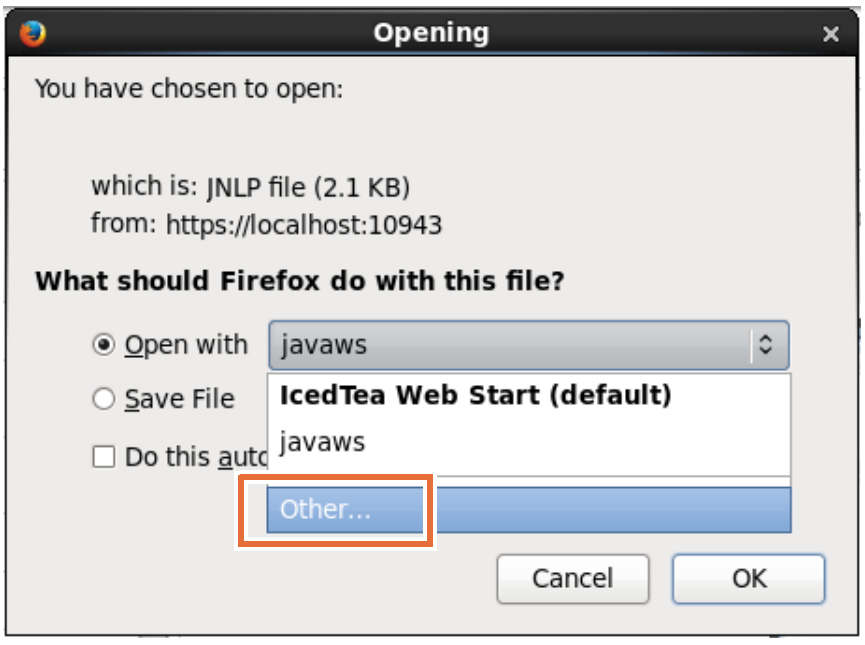


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. On the Run Tool window, click **Launch**.

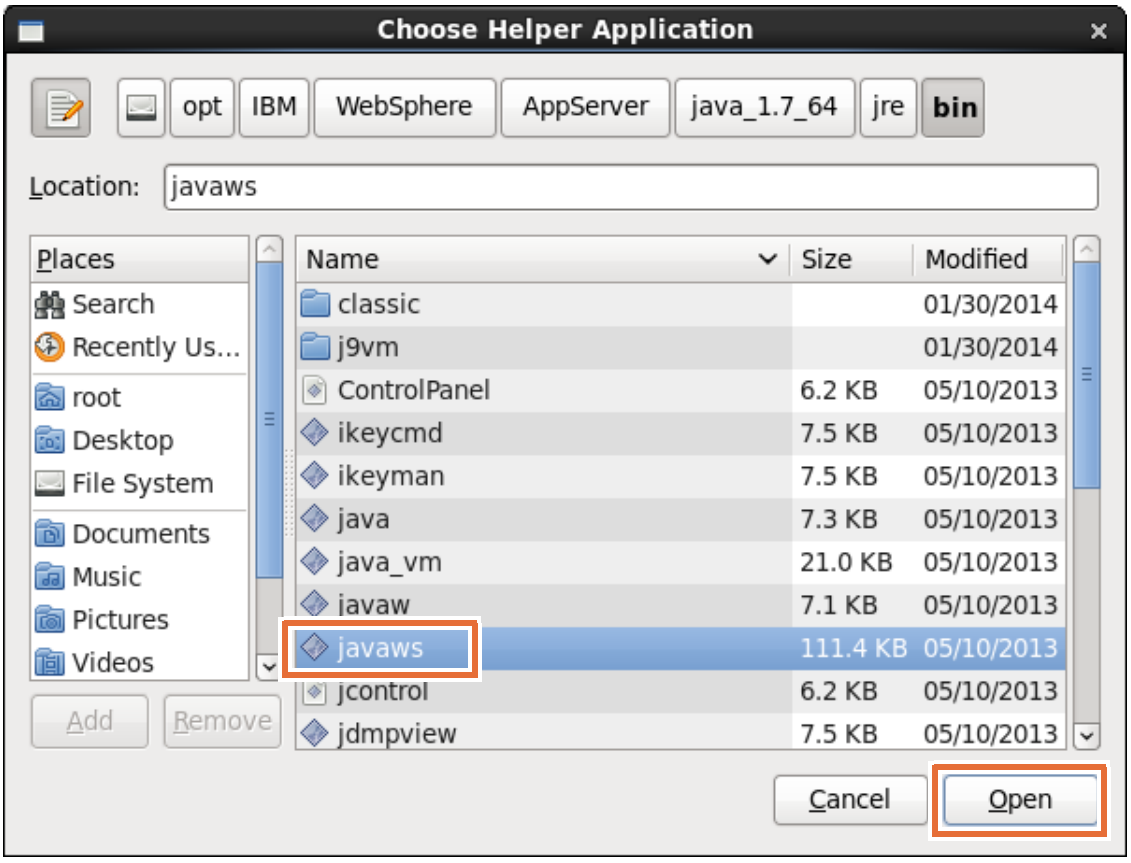


__ e. Expand the menu for the “Open with” option, and click **Other**.



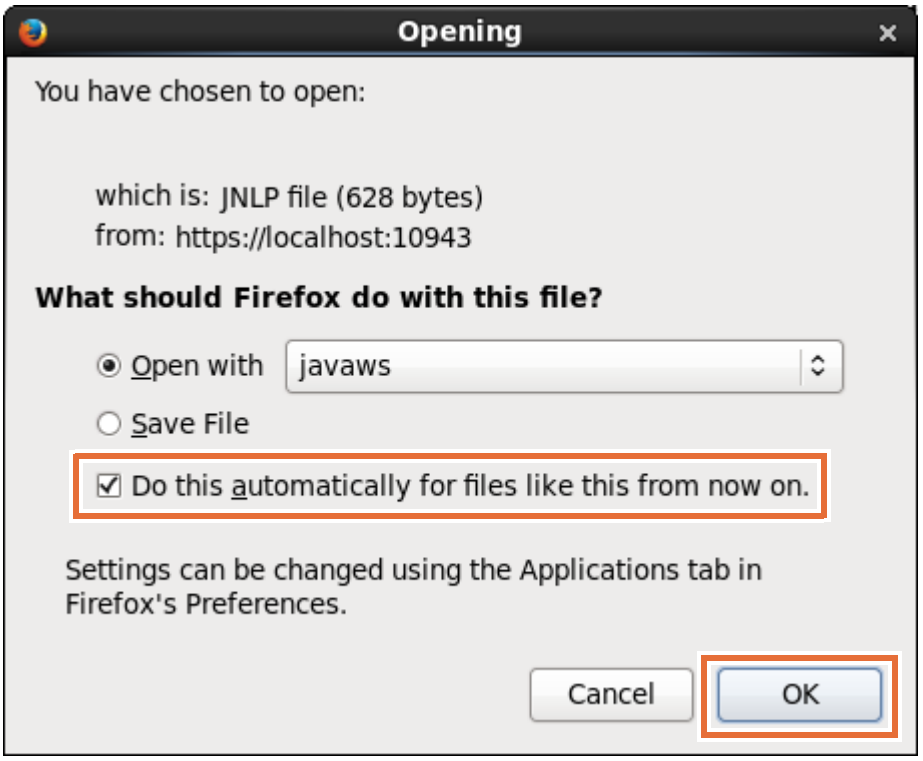
© Copyright IBM Corp. 2013
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Browse to `/opt/IBM/WebSphere/AppServer/java_1.7_64/jre/bin`, select **javaws** (Java Web Start) and click **Open.**



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. On the Opening pane, click the check box next to "**Do this automatically for files like this from now on.**"



__ h. Click **OK**.

Note

By selecting the option "**Do this automatically for files like this from now on**", you no longer see the Opening dialog box when you launch the desktop tools. All desktop tools are opened with the Java 7 version of Java Web Start (javaws)

Important

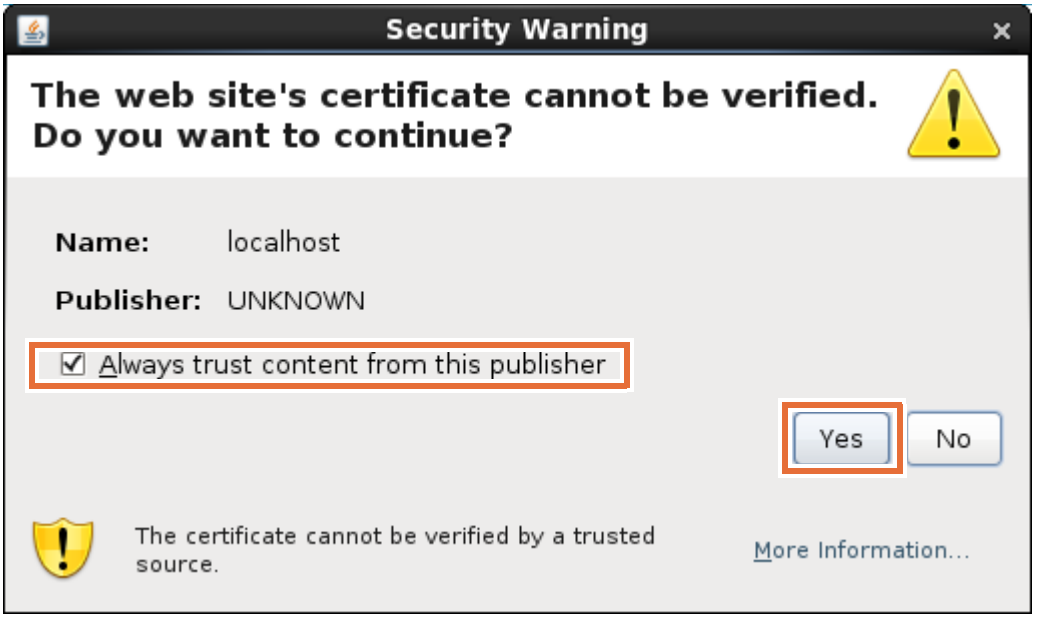
In the next several steps, you are starting a program from your browser. Most browsers consider starting a program to be a dangerous activity, and they should. As a result, several warning messages can appear. This exercise describes the warning messages that can appear. The warning messages can differ depending on your environment.

You might not see any of the following messages. If so, the certificates that are stored on your lab image are valid. In that case, proceed to step k.

© Copyright IBM Corp. 2013

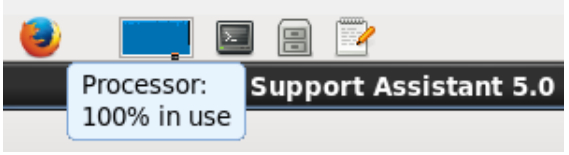
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ i. If you see the warning window, check **Always trust content from this publisher** and then click **Yes**.



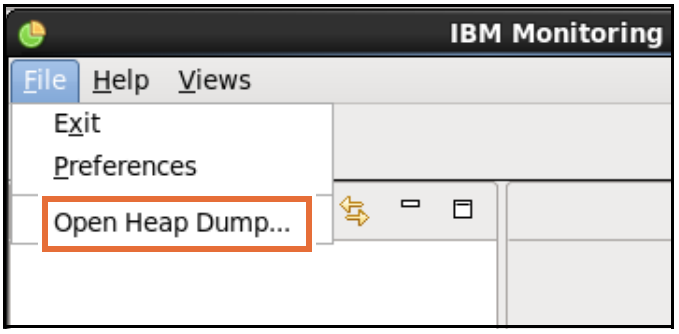
__ j. Another window warns about the digital signature. Check **Always trust content from this publisher** and then click **Run**.

__ k. The Memory Analyzer Desktop tool starts in its own window. It might take a few seconds for the tool to start. Monitor the processor status and make sure that the Memory Analyzer window is not hidden behind the web browser window.

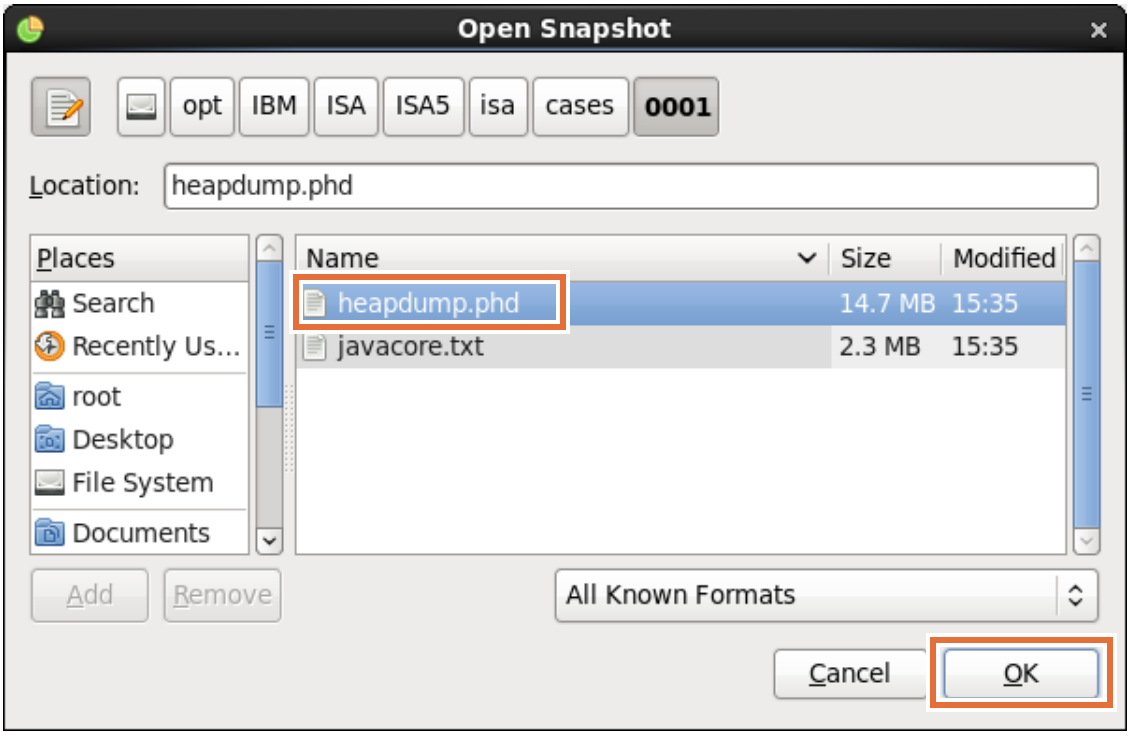


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ l. Click **File > Open Heap Dump**.



__ m. If the file browser does not open automatically to Case 0001, browse for `/opt/IBM/ISA/ISA5/isa/cases/0001`, and select the **heapdump.phd** file.



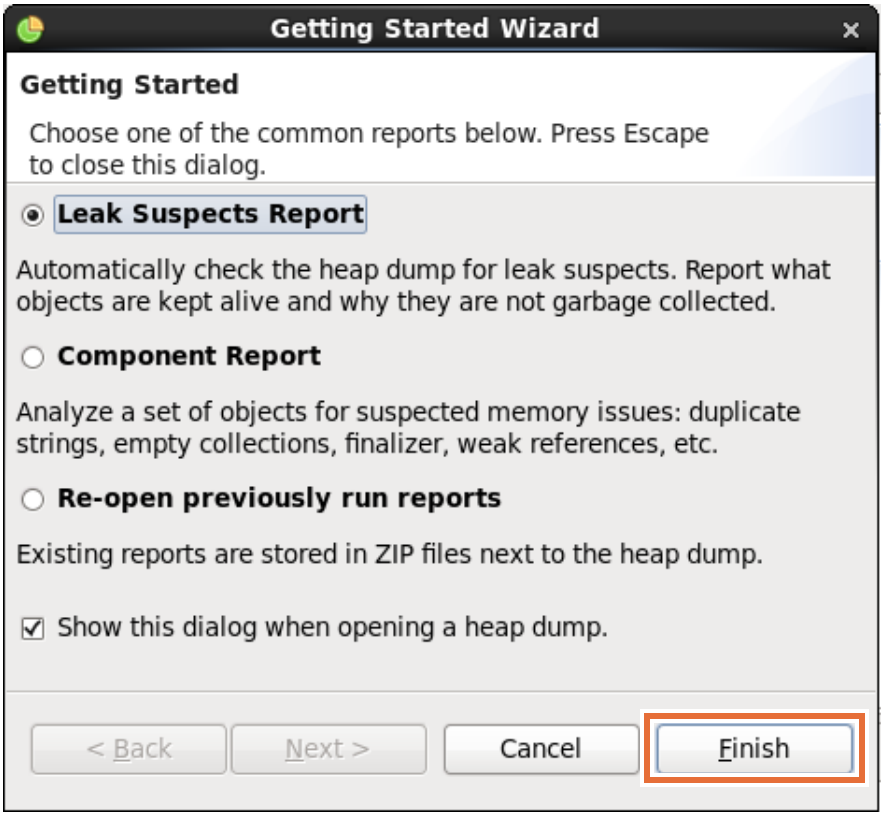
__ n. Click **OK**. By default, case files are stored in `/opt/IBM/ISA/ISA5/isa/cases`.

__ o. The parsing of the heapdump can take several minutes based on the size of the file. Wait until the Memory Analyzer completes its analysis of the heap dump file.

© Copyright IBM Corp. 2013

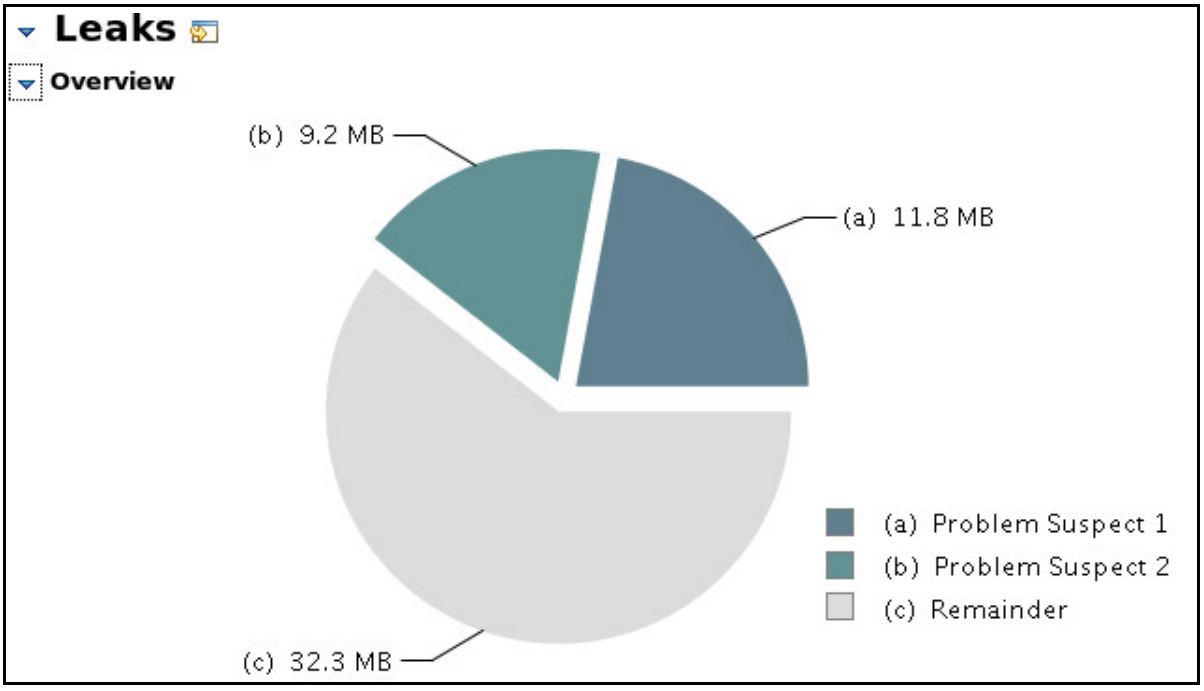
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ p. When you see the Getting Started Wizard, click the first option, **Leak Suspects Report**.



__ q. Click **Finish**. Wait until the report is prepared.

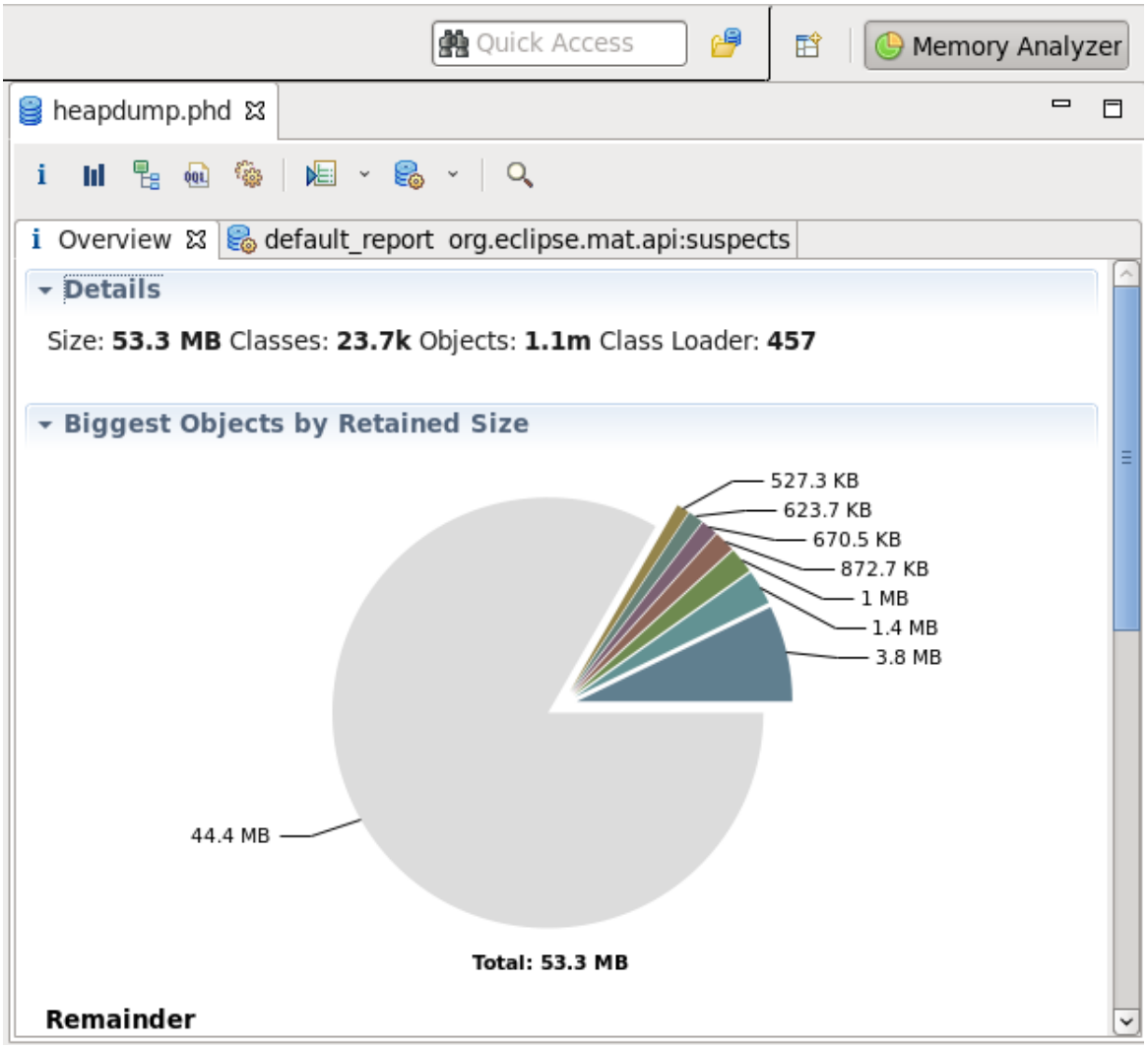
__ r. The report opens with the default report tab, which shows a pie chart of leak suspects.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. Explore the results of the Memory Analyzer.

__ a. As time permits, explore the different tabs, menus, and links in the Memory Analyzer.



__ b. For this exercise, it is sufficient to know how to launch a tool from IBM Support Assistant. Later in the course, you learn more about interpreting the results and analyses that different problem determination tools provide.

__ 5. Close the Memory Analyzer tool.

__ a. On the Memory Analyzer window, click **File > Exit**.

Section 4: Optional: Scan a case

IBM Support Assistant provides a service that scans all the files that are contained in a case. During that automated scan, any anomalies that are found in each file (such as unexpected errors or warnings) are collected as “Symptoms”. Then, they are searched in various IBM Knowledge Bases such as technotes and authorized program analysis reports (APARs) to find known potential solutions. The data that is collected from the scan and analysis is then displayed in the IBM Support Assistant views to help identify the root cause of the problem you are investigating.

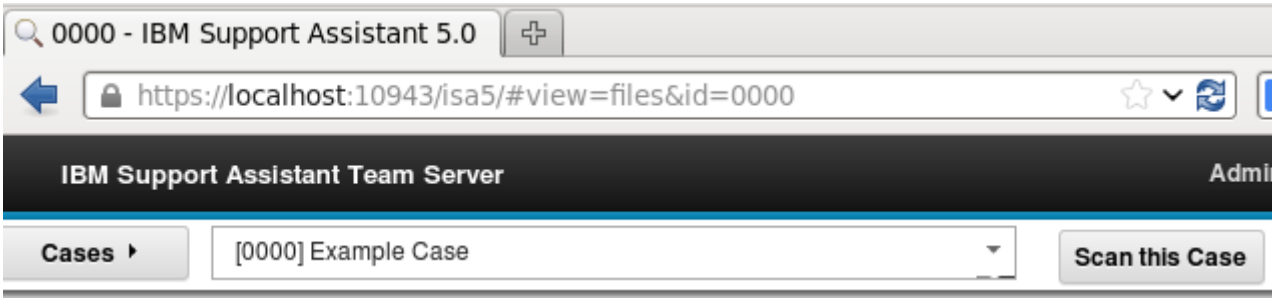
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

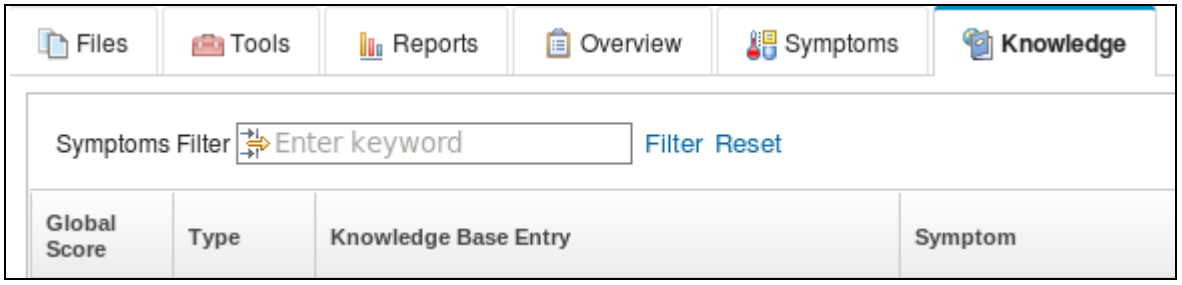
To scan a case, the IBM Support Assistant Team Server needs access to the Internet to search the IBM Knowledge base of APARs and technotes. If you are in an environment where Internet access is provided, you can complete this section. *If the Internet is not available, read the information box at the end of this section for alternative instructions.*

__ 1. Work with the Example Case.

__ a. From the IBM Support Assistant, select case **[0000] Example Case**.

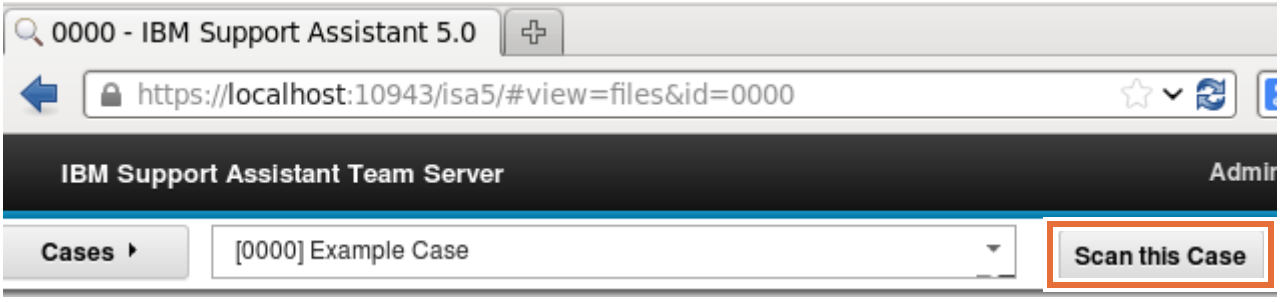


__ b. When a case is scanned, information about the case is written to the tabs: **Overview**, **Symptoms**, and **Knowledge**. Click each of these tabs now to see that they currently do not contain any information.



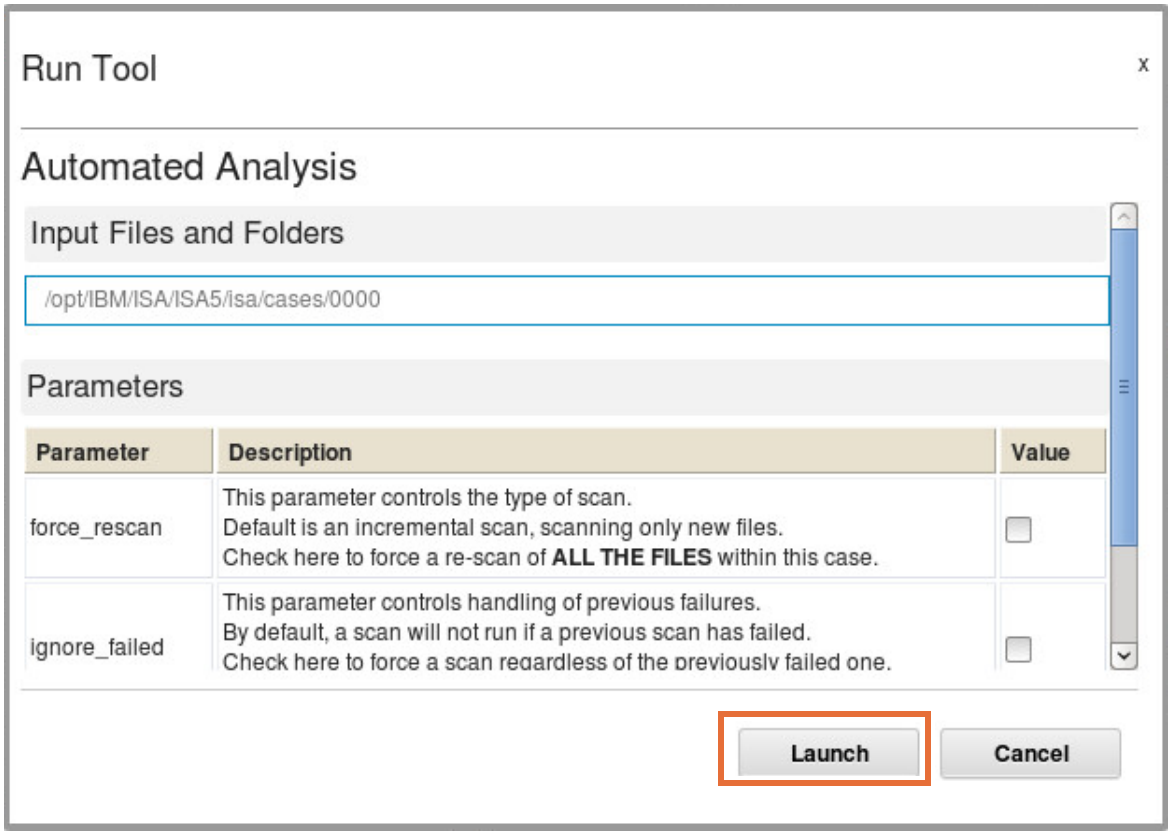
__ 2. Scan the Example Case.

__ a. Click **Scan this Case**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

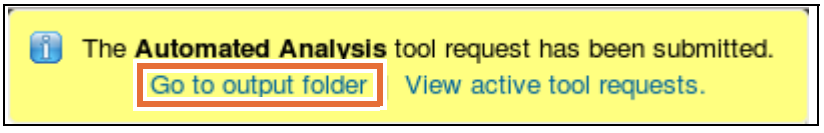
__ b. Before the scan begins, you are prompted to select **force_rescan** and **ignore_failed**. Since this scan is the first one you are running on this case, the options do not apply.



__ c. Click **Launch**.

__ d. After launching, you see the following notification:

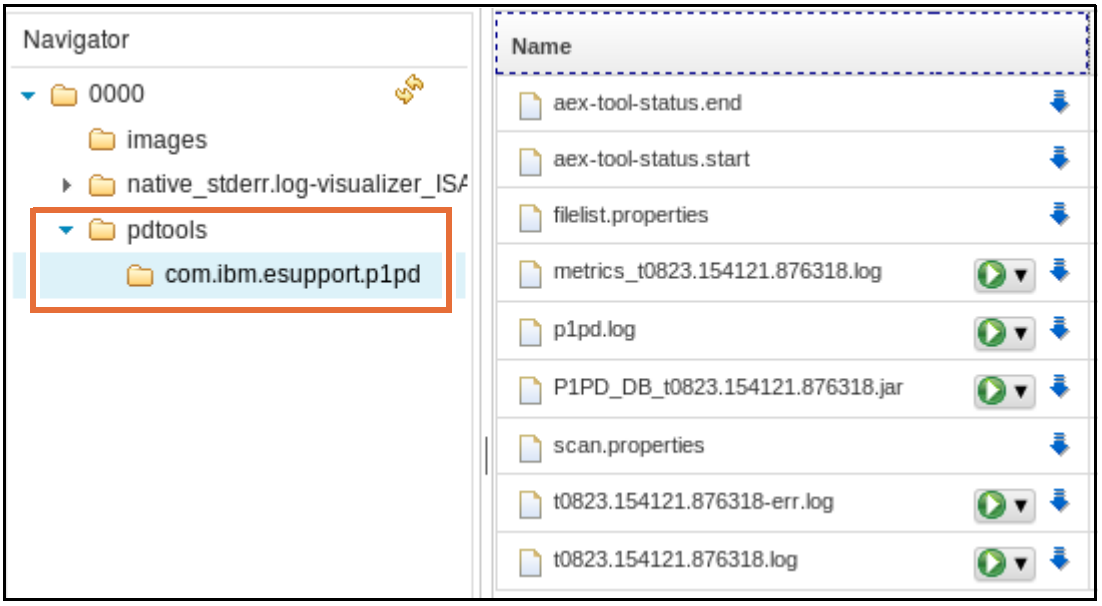
```
The Automated Analysis tool request has been submitted.
```



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

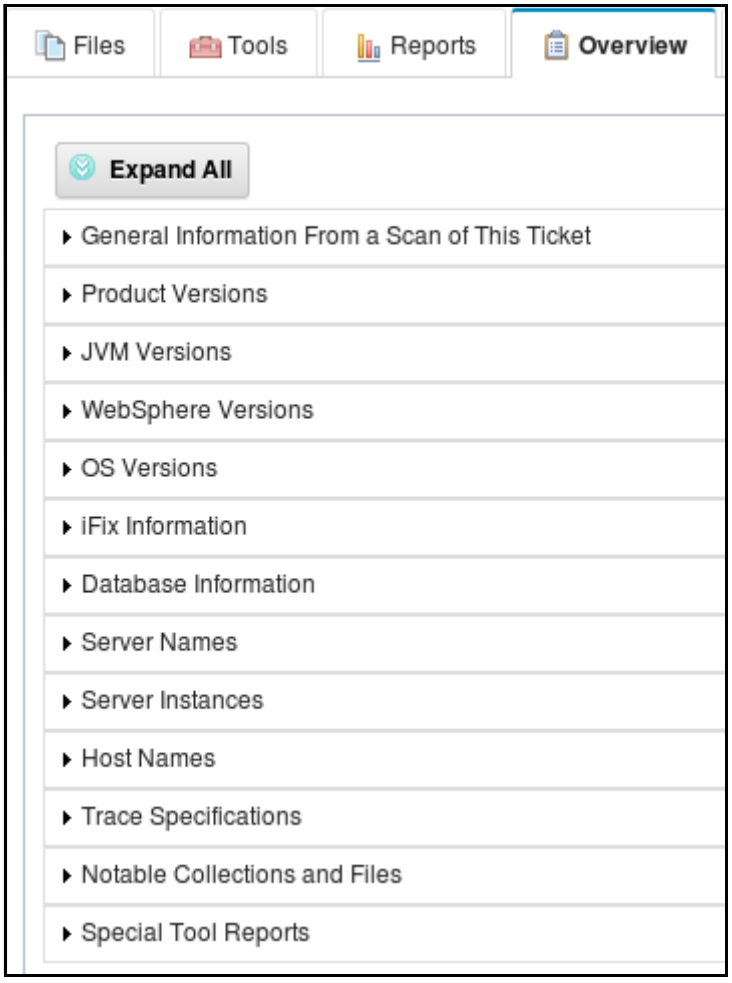
__ e. Click the link **Go to output folder**. The link takes you back to the Files tab. You see a new entry in the Navigator pane for the Automated Analysis files.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 3. Explore the tabs that contain the results of the scan.

__ a. Click the **Overview** tab. Click **Collapse All** to see a listing of all entries.

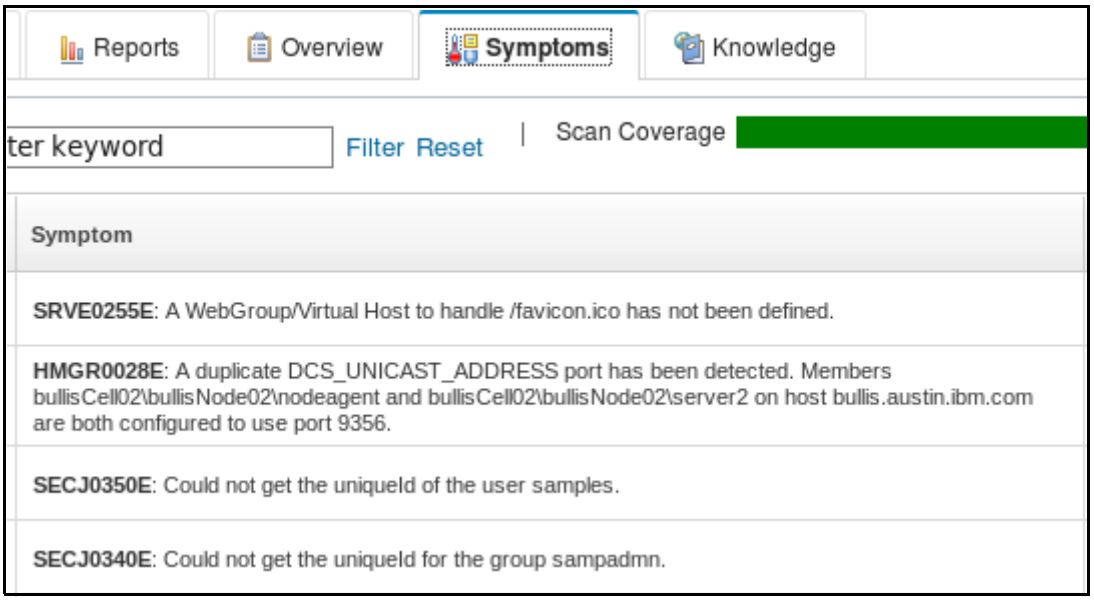


__ b. Expand any entry in the **Overview** tab to see more details. The data in this example is for illustrative purposes and was generated on a Windows Server that ran WebSphere V6.1.

 © Copyright IBM Corp. 2013

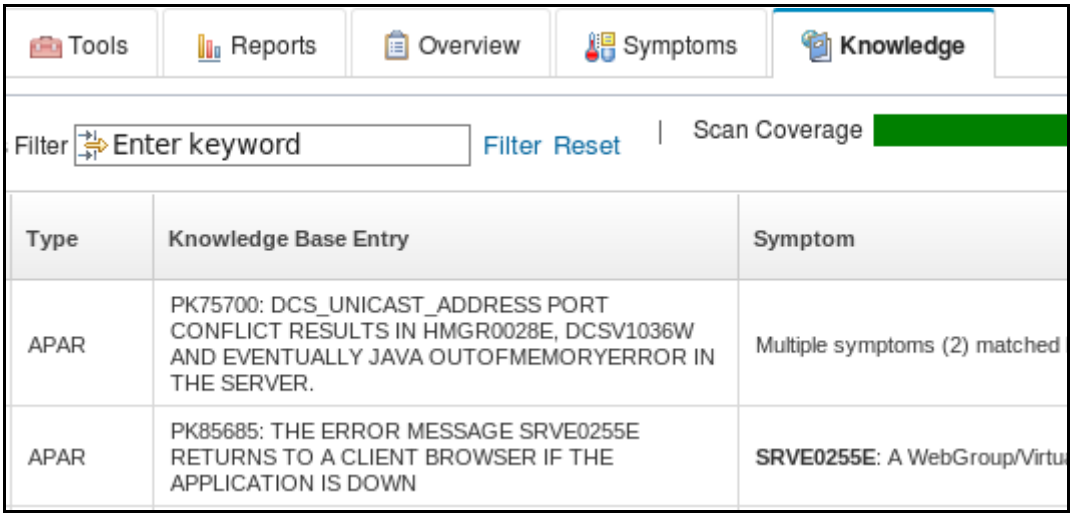
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Click the **Symptoms** tab.



__ d. The Symptoms tab lists error messages that the scan found in the diagnostic data for the case. It also lists the number of occurrences of the symptom, the number of items in the Knowledge Base that the symptoms match, and other useful information.

__ e. Click the **Knowledge** tab.



__ f. The Knowledge tab correlated the symptoms with Knowledge Base entries that can be of type APAR or technote.

__ g. Take time to explore each of the tabs in detail to see what type of problem determination the Automated Analysis provides.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

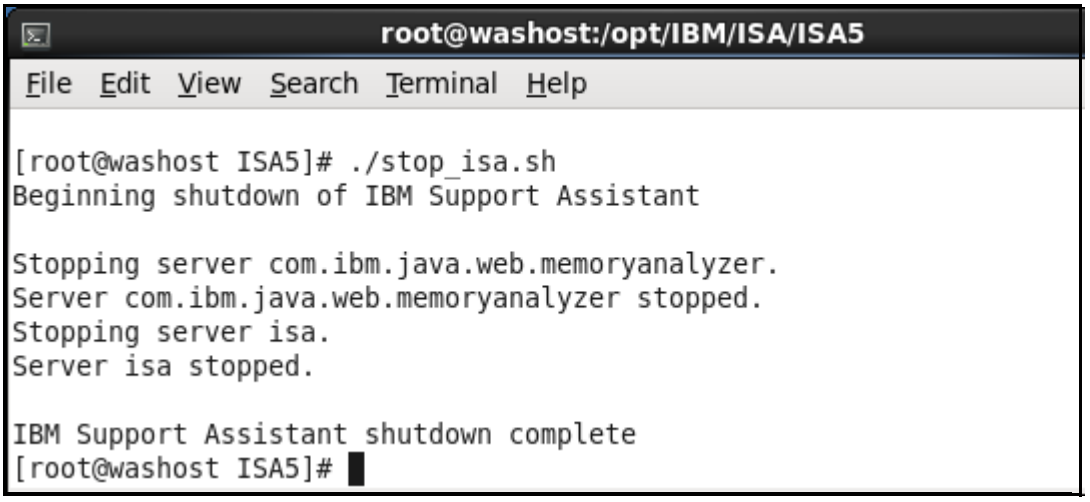
Alternative instructions for viewing the results of an Automated Analysis

1. Delete Case **0000**.
   a. Click **Cases** to open the Case Management dialog box.
   b. Select Case ID **0000** and click **Delete**.
   c. Close the web browser.
2. Use a file browser to navigate to `/usr/Student/ISA_Lab` and copy `0000_example_case_scanned.zip` to `/opt/IBM/ISA/ISA5/isa/Cases`
   a. Right-click `0000_example_case_scanned.zip` and select **Extract Here**.
   b. Open a new web browser to `http://localhost:10911/isa5`
   c. Case 0000 is now available with scan results. Select it from the case menu.
3. Return to Step **3** and view the scan results.

__ 4. After you are done exploring the Automated Analysis of the scanned case, shut down the IBM Support Assistant.

__ a. Close any web browsers windows and tabs that might still be open. From a Terminal window, change directory to: `/opt/IBM/ISA/ISA5`

__ b. Enter the command: `./stop_isa.sh`



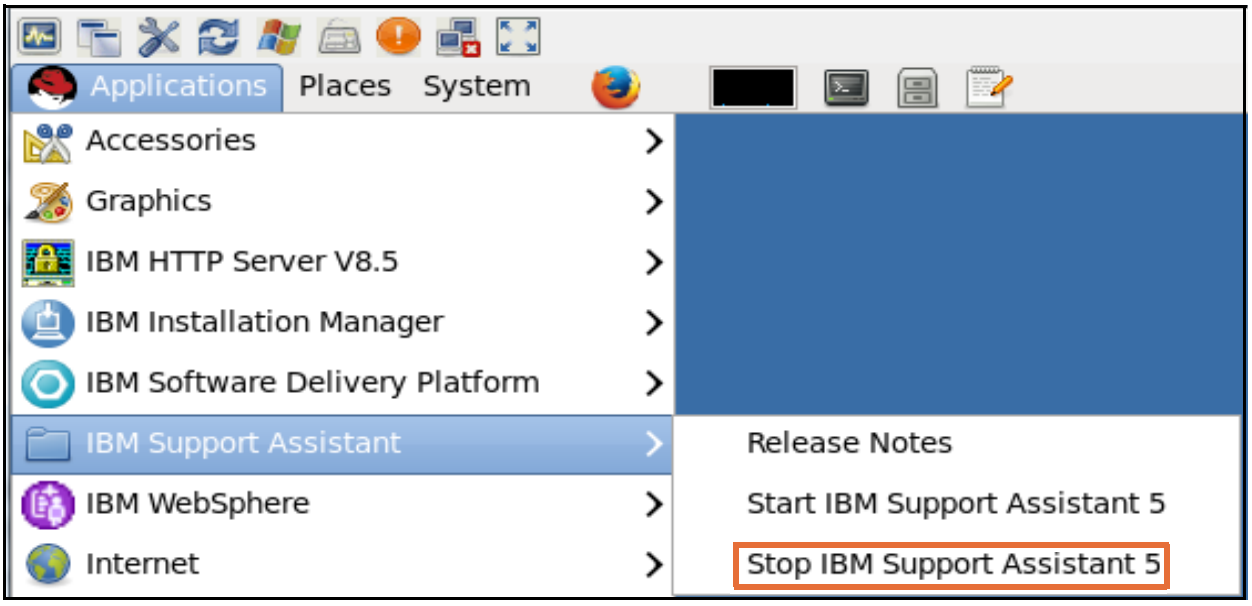
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

Start and stop shortcut

You can also stop the IBM Support Assistant from the Linux desktop by clicking Applications and selecting **IBM Support Assistant > Stop IBM Support Assistant 5**.



Section 5: Run the IBM Support Assistant Data Collector tool

Run the IBM Support Assistant Data Collector tool with the user ID for which you configured your WebSphere Server instance. Depending on what collection you are running, you are asked for more information to complete the data collection activities. A script might require more configuration information, information about the sequence of events that lead up to the problem you are dealing with, or your preferences about how it completes the collection. At each step, the choices are presented as numbered lists, and you input the number of your selection and press the Enter key. When input is required, prompts are displayed at which you enter your response and press the Enter key. You can find collection details for each WebSphere Application Server problem type in their corresponding MustGather documents.

__ 1. Start the IBM Support Assistant collector from the command line.

__ a. From a Terminal window, change directory to:

`<profiles_root>/profile1/bin`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the command: `./isadc.sh`

Accept the license agreement.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./isadc.sh

Starting IBM Support Assistant Data Collector in console mode...

Licensed Materials - Property of IBM Corporation Copyright, International
Business Machines, 2004, 2011.  IBM is a trademark or registered
trademark of IBM Corporation in the United States, other countries, or
both.

ApplicationServer: was85.2.0.3.20120525
IBM Support Assistant Data
Collector: 2.0.1.GA20120316-1743
Common Inventory Sub Agent:
6.3.0.20120123


Create the collection zip file at /root/washostCell-washostNode01-WAS-
ISADC.zip?
[1] Yes
[2] No
>
```

__ c. Accept the default location for the collection file by entering **1** at the prompt.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Decide what type of problem you need to collect data for, and enter the number of the type of collection that you want to run.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help

Enter the number of the IBM Support Assistant Data Collector option you
want to execute.
[1] Default Collection                [ distributed, IBM i, z/OS ]
[2] General...                        [ distributed, IBM i ]
[3] Administration...                 [ distributed, IBM i ]
[4] Security...                       [ distributed, IBM i ]
[5] Connectors...                     [ distributed, IBM i ]
[6] Containers...                     [ distributed, IBM i ]
[7] JDK...                            [ distributed ]
[8] Runtime...                        [ distributed, IBM i ]
[9] HTTP...                           [ distributed, IBM i ]
[10] Service Oriented Architecture... [ distributed, IBM i ]
[11] Intelligent Management Pack...   [ distributed, IBM i ]
[12] Quit
>
```

__ e. The options with an ellipsis (...) have subcollectors for different troubleshooting areas. Enter **4** for security to see what other security issues have collectors.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
[10] Service Oriented Architecture... [ distributed, IBM i ]
[11] Intelligent Management Pack...   [ distributed, IBM i ]
[12] Quit
> 4

Enter the number of the IBM Support Assistant Data Collector option you
want to execute.
[1] Security Problem
[2] JSSE/SSL Problem
[3] Java 2 Security Problem
[4] Problem running as Non-Root
[5] Return to Previous Menu
[6] Quit
>
```

__ f. Notice the different security-related problems such as JSSE/SSL and Java 2 for which you can choose to run separate collectors. Enter 5 to return to the previous menu.

__ g. Try entering the number for other collectors that have an ellipsis, such as Connectors..., JDK..., and HTTP..., to see the specific areas for which they provide collectors.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. If you choose to run a specific collector, you are prompted interactively to answer questions about your environment, provide profile and server names, administrator name, and password. You might also be asked whether you want to restart a server and reproduce a problem. Finally, the collector gathers all the diagnostic data and archives it in a compressed file, and stores it in the location you chose when you started the IBM Support Assistant Data Collector. For this exercise, you are ***not*** going to run a collector, since it can take a long time. A previously run collection is available for you to examine.

   __ a. Use the file browser to navigate to: `/usr/Student/ISA_Lab/ISADC`

   __ b. Right-click the file that is called `washostCell-washostNode01-WAS-ISADC.zip` and select **Open with the Archive Manager**.

   __ c. Examine the data and logs that were collected. Close Archive Manager and the file browser when you are finished.

__ 3. Quit the IBM Support Assistant Data Collector.

   __ a. Return to the main menu, and enter `12` to quit.

End of exercise

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

In this exercise, you learned how to start and stop the IBM Support Assistant Team Server. You logged in to the Administration for the IBM Support Assistant Team Server and examined the tools for installing, updating, and uninstalling problem determination tools. You also learned how to run report generator tools and examine the reports and use the Case Manager to create a case and add diagnostic data. You used the Tools tab to run interactive desktop tools. You used Automated Analysis to scan a case and examine the results. Finally, you ran the IBM Support Assistant Data Collector tool, which is a tool that runs outside of the IBM Support Assistant to help gather diagnostic data.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 2. Gathering diagnostic data

What this exercise is about

This exercise covers different types of tools that are built into the administrative console that can be used for problem determination. You explore some of the tools available. You also look at diagnostic data that is available in log and trace files. Finally, you use different tools to examine views of the JNDI namespace.

What you should be able to do

At the end of this exercise, you should be able to:

- Configure basic (legacy) logging and examine server log files
- Enable HPEL logging for an application server
- Enable tracing on an application server component and read a trace output log
- Examine logs and trace data by using the HPEL Log Viewer
- Enable and configure Cross Component Trace (XCT)
- Use the administrative console troubleshooting section to view runtime messages and configuration problems
- Enable and use diagnostic providers in the administrative console
- Examine FFDC logs
- Use the Classloader viewer in the administrative console
- Compare namespaces from different contexts by using the dumpNameSpace tool and the name server MBean

Introduction

In this exercise, you learn how to locate and view log files for the WebSphere Application Server. You learn how to enable HPEL logging for an application server. You also learn how to enable tracing for an application server by using the administrative console. You examine other sources of diagnostic data about class loaders by using the Classloader viewer, and compare namespaces by using the dumpNameSpace tool and name server MBean.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Requirements

To do this exercise, you must have a working WebSphere Application Server **server1**, administrative console, and a running default application and PlantsByWebSphere application, which are installed on **profile1**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Working with log files of the application server

In this section, you examine the configuration options for Basic mode (legacy) logging.

__ 1. Edit the /etc/hosts file and add the IP address of your lab image. This address is necessary for Intelligent Management to work properly.

- __ a. Open a terminal window, and enter the command ifconfig -a
- __ b. Record the network address for eth0 here_____
- __ c. Use a text editor to open the /etc/hosts file.
- __ d. Add a new first line with the network address and the host name.

  For example:

  `172.16.80.111  washost`
- __ e. Save and close the file.

__ 2. Start the deployment manager.

- __ a. In a command prompt change directory to: `<profiles_root>/profiles/Dmgr/bin`
- __ b. Enter the command: `./startManager.sh`

__ 3. Start the node agent and server1.

- __ a. In a command prompt change directory to:

  `<profiles_root>/profiles/profile1/bin`
- __ b. Start the node agent with the command: `./startNode.sh`
- __ c. Wait for the node agent to start successfully.
- __ d. Start server1 with the command: `./startServer.sh server1`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. Open the administrative console.

__ a. Open a Firefox web browser and specify the web address:

`http://washost:9060/ibm/console`

__ b. Log in with as follows:

- User ID: `wasadmin`
- Password: `web1sphere`

__ 5. You can change the location, name, and other settings of log files by using the administrative console.

__ a. In the navigation tree, select **Troubleshooting > Logs and trace**. In the pane on the right, click **server1**.

Information

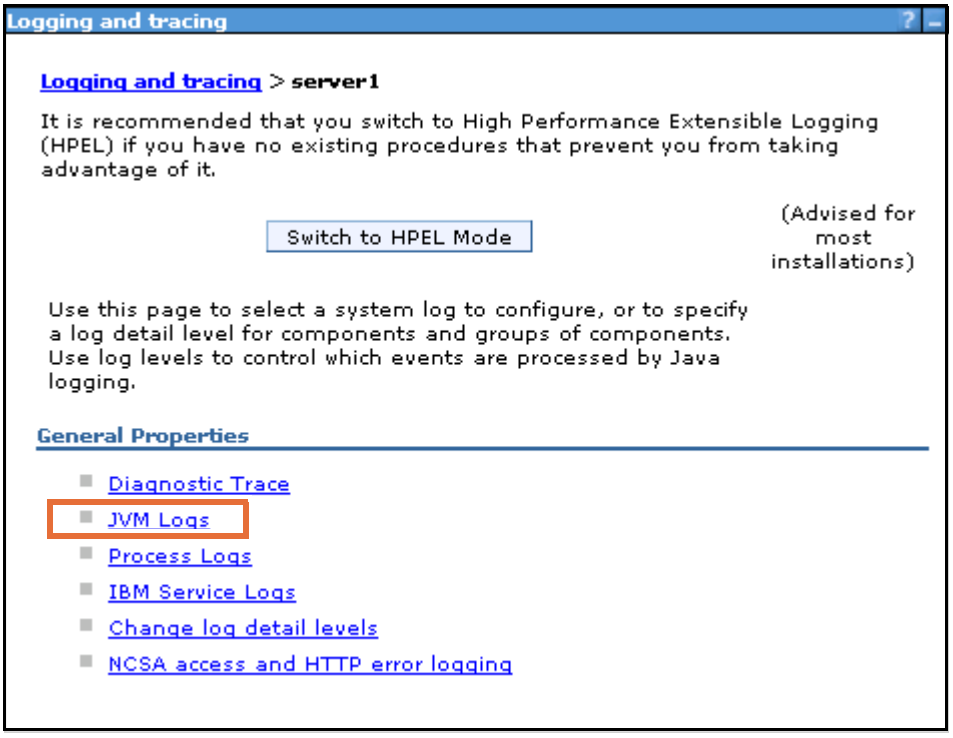
You can also reach the configuration area for Logging and Tracing by selecting **Servers > WebSphere application servers > server1**. Click **Logging and Tracing** under the Troubleshooting section.

__ 6. Change the number of historical files and set the maximum size of the log file for `System.out`. The number of historical files grows from zero to the value of the maximum number of historical files field. The next rollover deletes the oldest historical file.

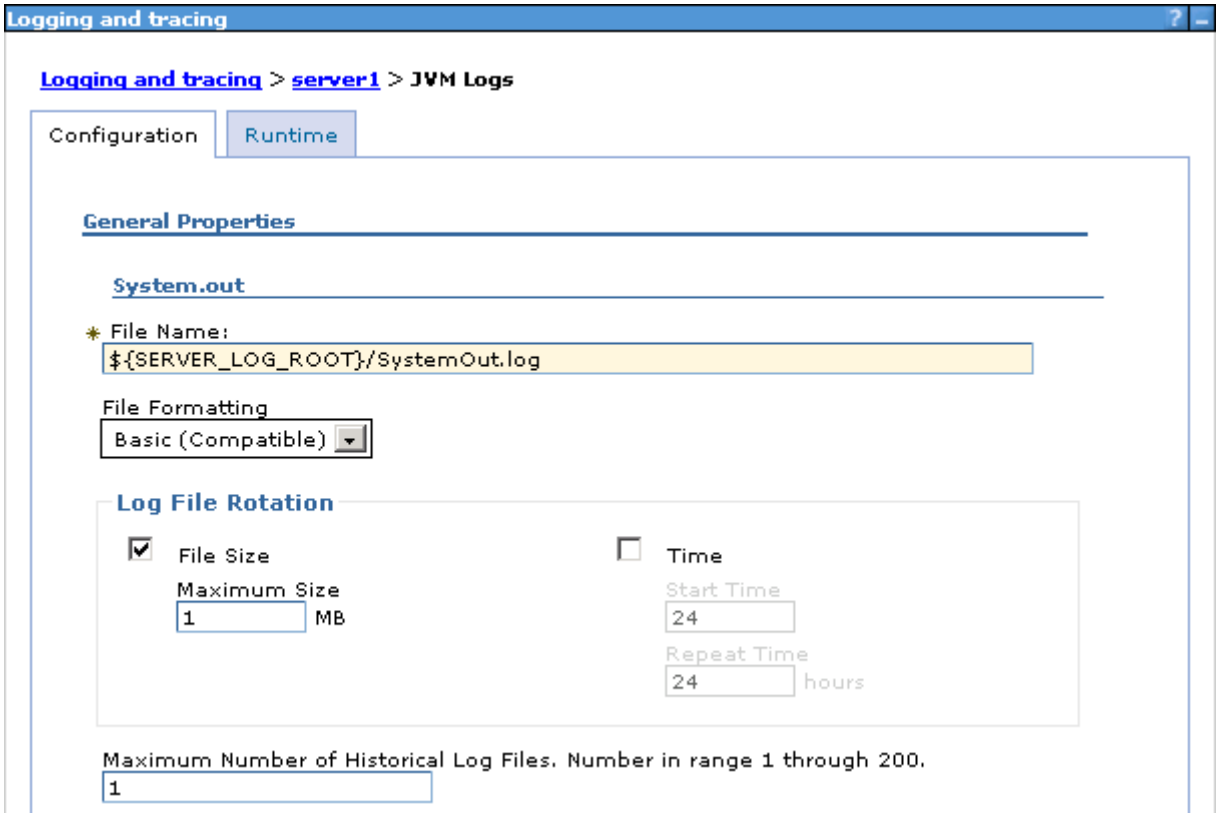
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ a. Select **JVM Logs**.



__ b. You can view and modify settings from the Logging and tracing panel for the `System.out` and `System.err` logs.

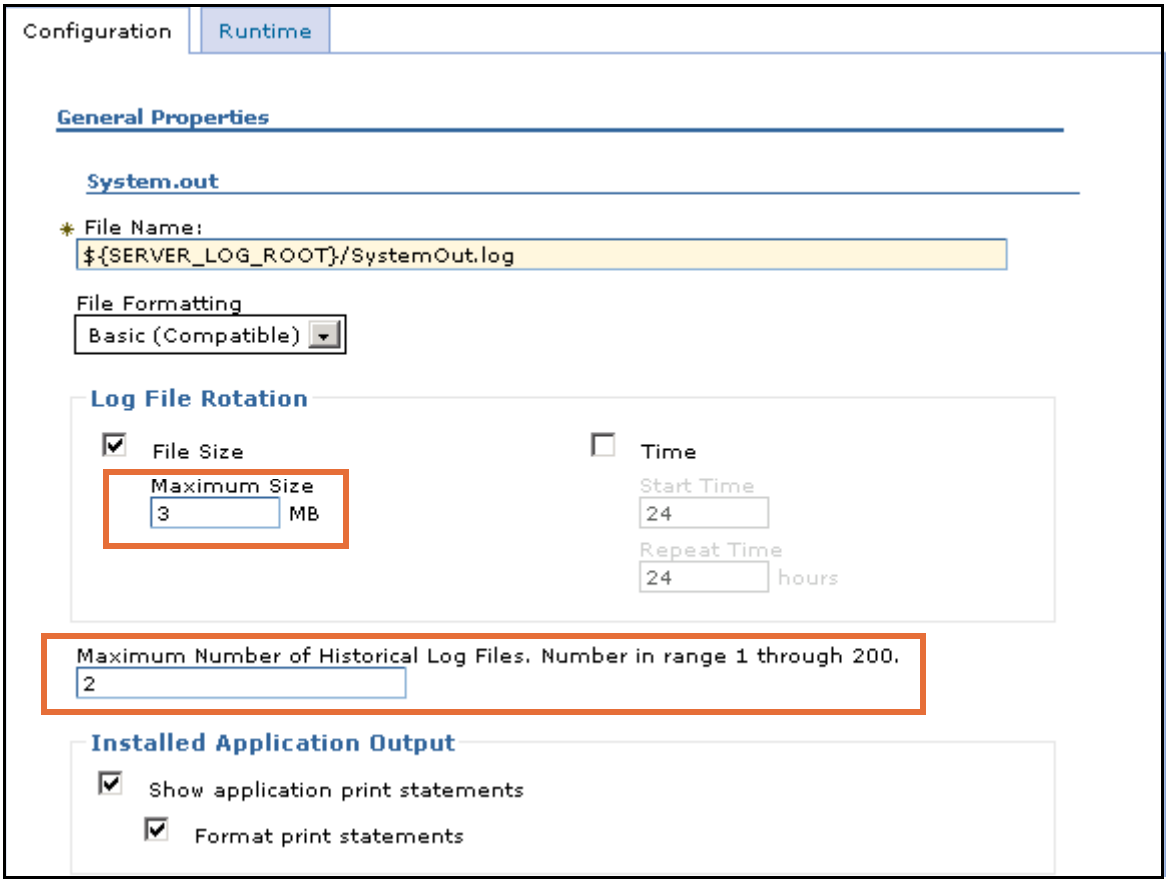


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

Use this page to view and modify the settings for the Java virtual machine (JVM) `System.out` and `System.err` logs for a managed process. The JVM logs are created by redirecting the `System.out` and `System.err` streams of the JVM to independent log files. The `System.out` log is used to monitor the health of the running application server. The `System.err` log contains exception stack trace information that is used for problem analysis. One set of JVM logs exists for each application server and all of its applications. JVM logs are also created for the deployment manager and each node agent. Changes on the Configuration panel apply when the server is restarted. Changes on the Runtime panel apply immediately.

__ c. Under **General Properties** for `System.out` set the **Maximum Size** to **3 MB** and the **Maximum Number of Historical Log Files** to **2**.



__ d. Click **OK**.

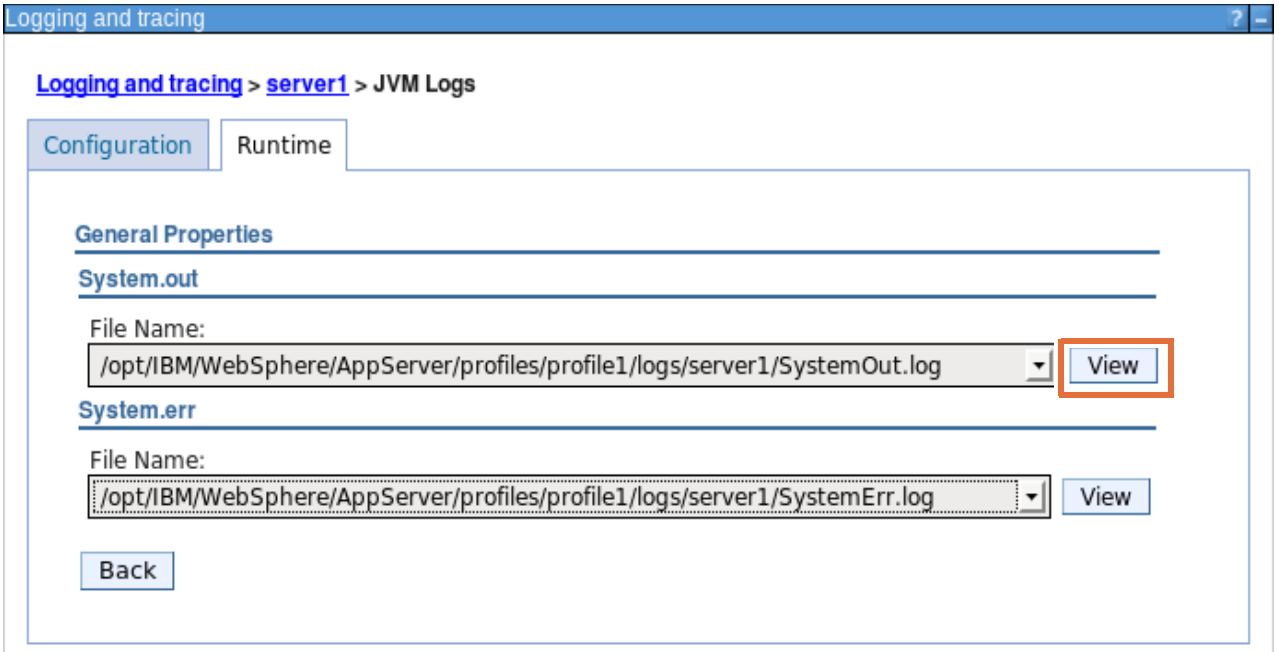
__ e. **Save** the changes to the master configuration.

__ 7. View `SystemOut.log` and `SystemErr.log` for server1 by using the administrative console.

© Copyright IBM Corp. 2013

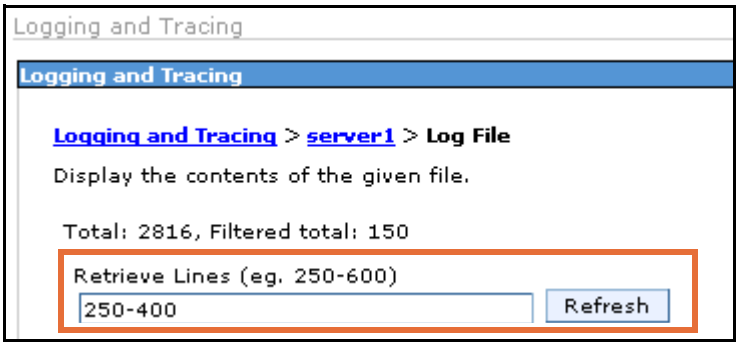
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ a. Select **Troubleshooting > Logs and trace > server1 > JVM Logs** and select the **Runtime** tab.



__ b. Click **View** to the right of the **File Name** field for `System.out`.

__ c. The default is to retrieve 250 lines in one step. You can specify the range of lines to retrieve at the top of the Logging and Tracing window. Retrieve lines `250-400` by typing `250-400` and clicking **Refresh**.



__ d. Notice that several lines from the log file are displayed. Scroll down to view the log records. Click **OK** when finished.

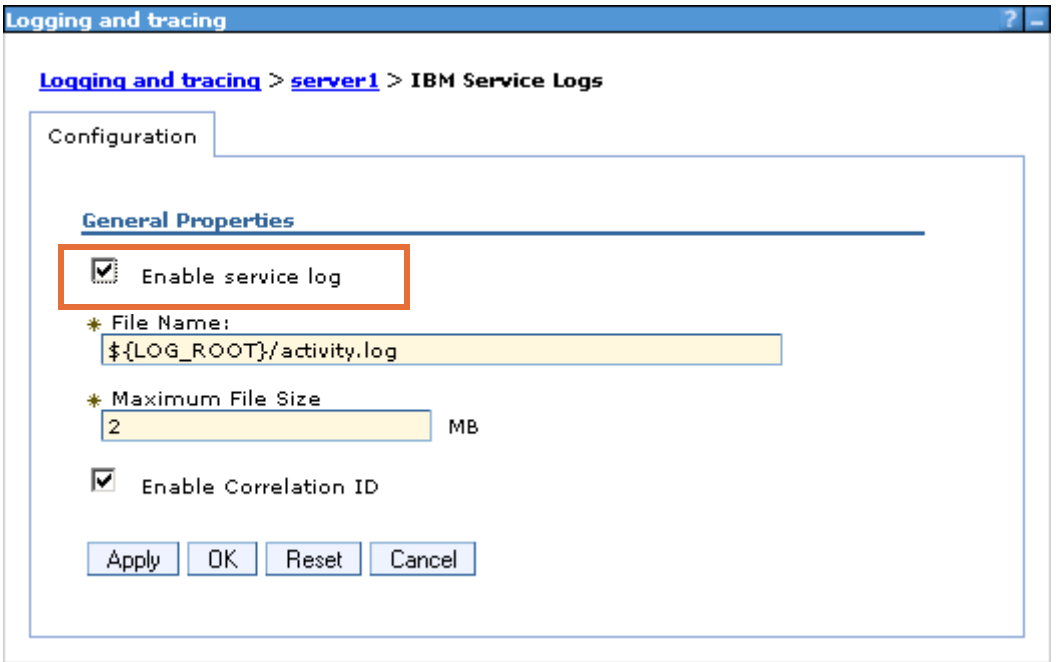
Information

You can also navigate to the `<profile_root>/profile1/logs/server1` folder to view the logs by using a text editor. Using an editor is preferable since you can use the search features of your text editor.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 8. Configure the IBM service logs by using the administrative console. Unlike the JVM logs, the IBM service logs cannot be viewed within the administrative console. You must use a tool such as the Log Analyzer, which is available in the IBM Support Assistant.

__ a. Select **Troubleshooting > Logs and trace > server1 > IBM Service Logs**.



__ b. Check the box for **Enable service log**.

Note

In WebSphere Application Server Version 7 and earlier, the service log is enabled by default. In WebSphere Application Server Version 8 and later, however, the service log is disabled by default. If you do not intend to enable HPEL for a server, then you can enable the IBM service log to be consistent with earlier versions.

Information

The IBM service log contains both the application server messages that are written to the `System.out` stream and special messages that contain extended service information that you can use to analyze problems. One service log exists for all Java virtual machines (JVMs) on a node, including all application servers and their node agent, if present. A separate activity log is created for a deployment manager in its own logs directory. The IBM Service log is maintained in a binary format. Use the Log Analyzer or Showlog tool to view the IBM service log.

To find the value for $(LOG_ROOT), you can look to **Environment > WebSphere Variables**. The name of the service log is `activity.log`, but the name can be changed along with its location in the file system. Maximum file size can be set, and you can enable or disable a correlation ID. You

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

can use the correlation ID to correlate activity to a particular client request. You can also use it to correlate activities on multiple application servers, if applicable.

__ c. Click **OK**.

__ d. **Save** changes to the configuration.

Section 2: Enable and configure HPEL

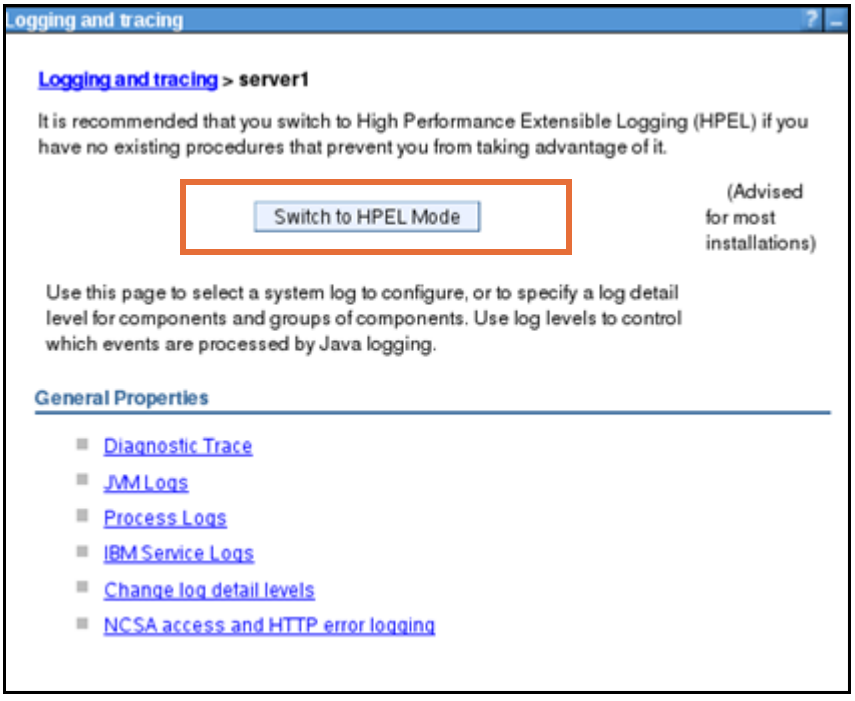
High Performance Extensible Logging (HPEL) is a new mode of logging and tracing in WebSphere Application Server V8. To take advantage of this new log and trace framework, HPEL mode must be enabled. After HPEL mode is enabled, the JVM logs (typically `SystemOut.log` and `SystemErr.log`), the trace log (typically `trace.log`) and the service log (typically `activity.log`) are no longer written to. Instead, log and trace content is written to a log data or trace data repository in a proprietary binary format and, if configured, to a text log file. To gain the largest performance benefit of HPEL, disable writing results to a text file. A log viewing tool, Log Viewer, is provided to allow for viewing, filtering, monitoring, and formatting the log and trace data in the repositories.

In this section, you enable HPEL mode for server1. Then, you explore and modify the log and trace configurations.

__ 1. Enable HPEL for server1.

__ a. Navigate to the Logging and tracing panel for server1 by clicking **Troubleshooting > Logs and trace > server1**.

__ b. Enable HPEL by clicking **Switch to HPEL Mode**.



__ c. Click **Save** to save the configuration.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Restart **server1**. It is necessary to restart a server for the HPEL mode to become effective.

__ a. Navigate to **Servers > Server Types > WebSphere application server**.

__ b. Select **server1** and click **Restart**.

__ 3. Configure HPEL for server1.

__ a. Navigate to the **Logging and tracing panel** for server1 by clicking **Troubleshooting > Logs and trace > server1**.

Logging and tracing

Logging and tracing > server1

Use this page to select a system log to configure, or to specify log detail levels for components and groups of components.

General Properties

Configure HPEL logging

Directory	/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1
For cleanup, delete records older than	Disabled
For cleanup, maximum size of logs	50 Megabytes

Configure HPEL trace

Directory	/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1
For cleanup, delete records older than	Disabled
For cleanup, maximum size of trace	50 Megabytes

Configure HPEL text log

Current status:	Enabled
Directory	/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1
For cleanup, delete records older than	Disabled
For cleanup, maximum size of text log	50 Megabytes

__ b. In the general properties section, you can see the current (default) configuration for the three HPEL repositories. Each has a directory location, and cleanup options for age and size of log files. Notice that the HPEL text log has a status of Enabled. In the next step, you disable the text log.

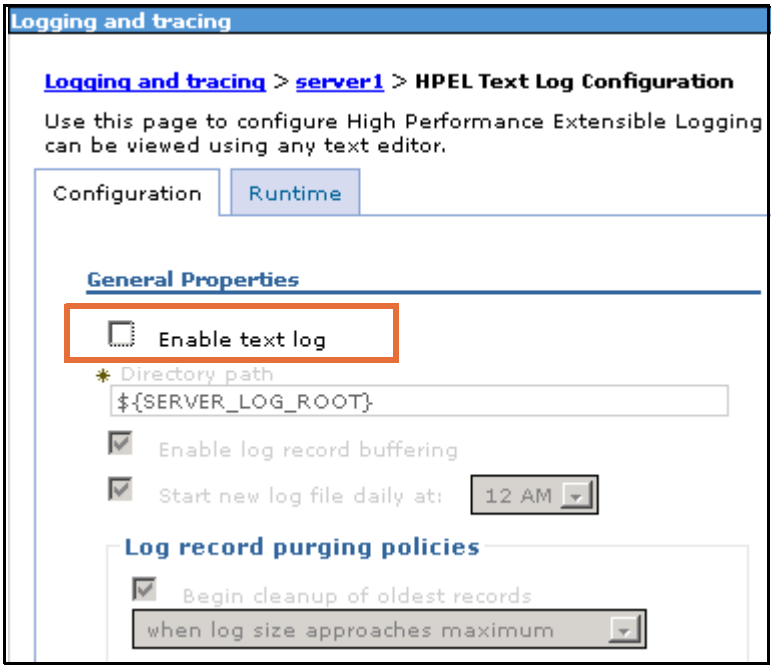
__ 4. Disable the HPEL text log.

__ a. In the General Properties section, click the link **Configure HPEL text log**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Clear the box for **Enable text log**.



__ c. Click **OK**.

__ d. Click **Save** to save the configuration.

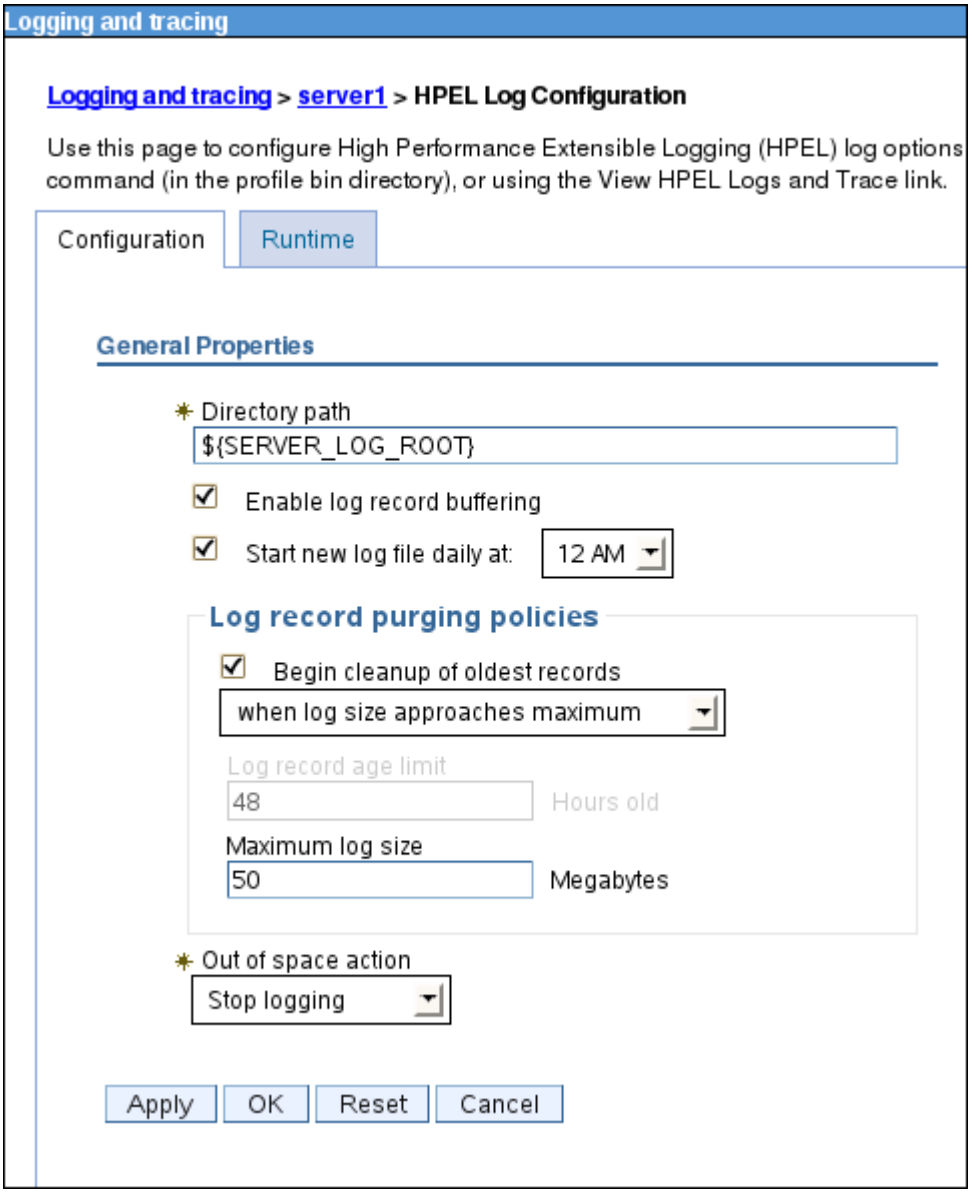
__ e. It is necessary to restart the server for this change to take effect. However, wait until you make some other configuration changes before you restart server1.

__ 5. Explore the configuration for HPEL logging.

__ a. Navigate to **Troubleshooting > Logs and trace > server1**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the General properties tab, click the link **Configure HPEL logging**.



__ c. Notice the different options for configuring HPEL logging. Also, notice that log buffering is enabled. Since buffering improves performance, it is good to keep it enabled.

__ 6. Modify the HPEL log configuration.

__ a. Change **Maximum Log Size** from **50** to **20** MB.

__ b. Click **OK**.

__ c. Click **Save** to save the configuration.

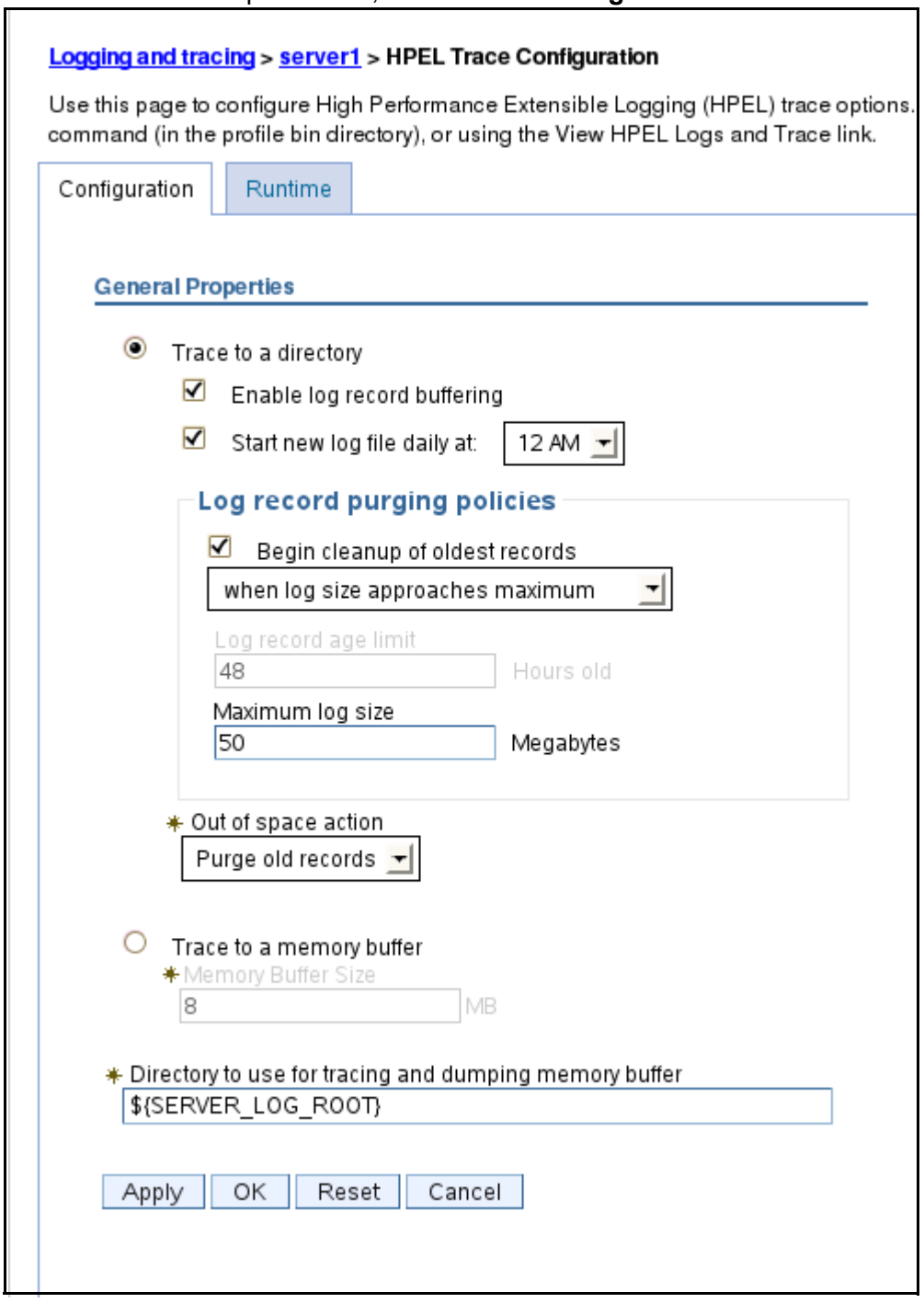
__ 7. Explore the configuration for HPEL tracing.

__ a. Navigate to **Troubleshooting > Logs and trace > server1**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the General Properties tab, click the link **Configure HPEL trace**.



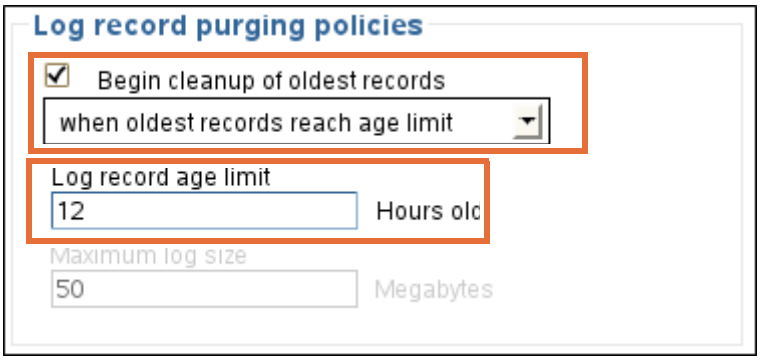
__ c. Notice that you have options to trace to a log directory or memory buffer. The default is to trace to a log directory.

__ 8. Modify the HPEL Trace configuration.

__ a. Under **Log record purging policies**, to configure when to begin cleanup of oldest records, use the drop-down menu to select **when oldest records reach age limit**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Next, set **Log record age limit** to **12** hours.



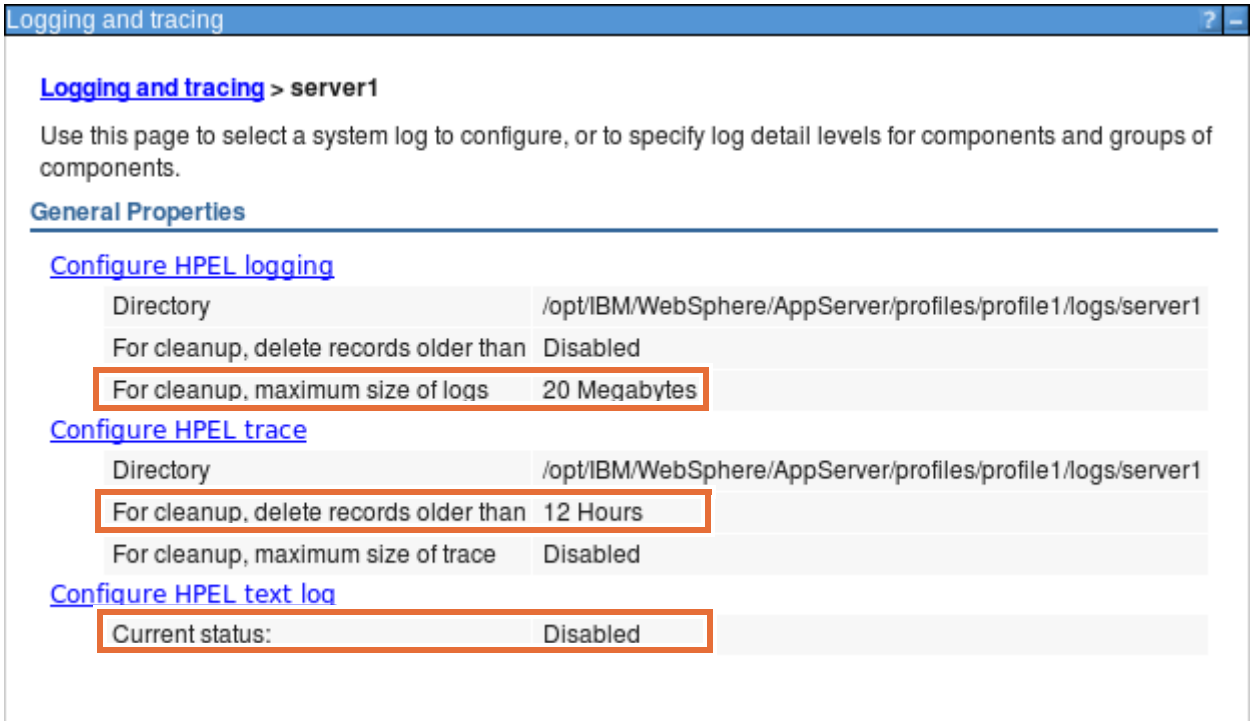
__ c. Click **OK**.

__ d. Click **Save** to save the configuration.

__ e. From the administrative console, restart **server1** so that the new configuration is in effect.

__ 9. Verify new HPEL configurations.

__ a. Click **Troubleshooting > Logs and trace > server1**.



__ b. Verify that all the configuration changes that you made are now in effect.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 3: Use the Log Viewer in the administrative console to examine log data and trace data

In this section, you use the Log Viewer in the administrative console to examine the log messages for an application server. You use various filtering functions to customize which log records are displayed.

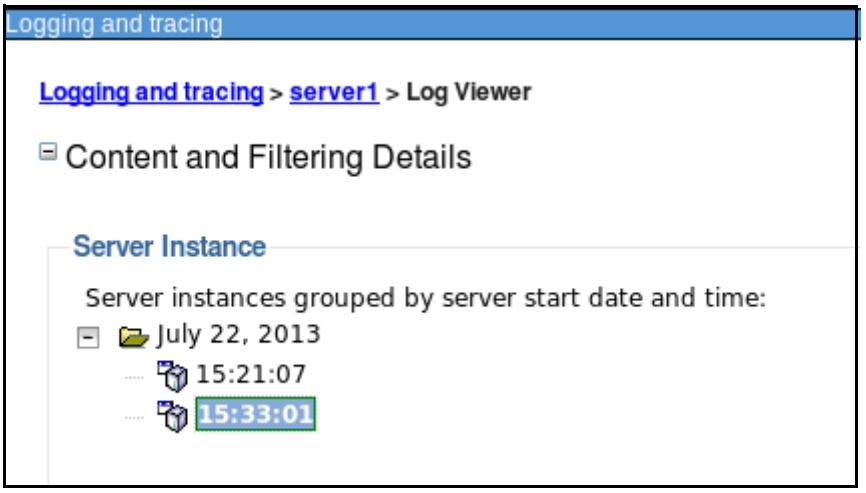
__ 1. Navigate to the Log Viewer for server.

__ a. Click **Troubleshooting > Logs and trace > server1 > View HPEL Logs and Trace**.

__ 2. Select the latest instance of the application server.

__ a. Expand **Content and Filtering Details**.

__ b. Expand the **Server Instance** tree, and make sure that the latest instance of server is selected (highlighted).



Note

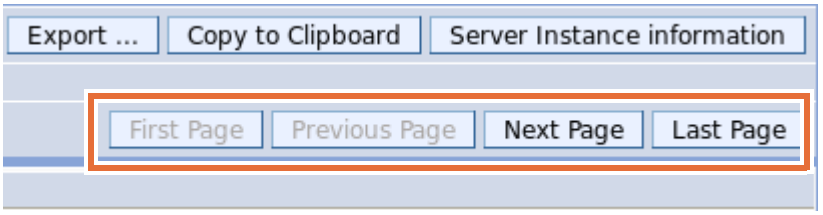
Since you recently enabled HPEL for server1, you see one instance of the server log repository. An instance is created for each new start of the server and designated with a time stamp. In this example, which uses the default configuration, the instances for each day are stored in a folder that is designated with the date. Any instance can be viewed by selecting it, and the log records are displayed in the Log Viewer.

__ c. Collapse the **Content and Filtering Details** tree.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 3. Use the Log Viewer to explore the log records.

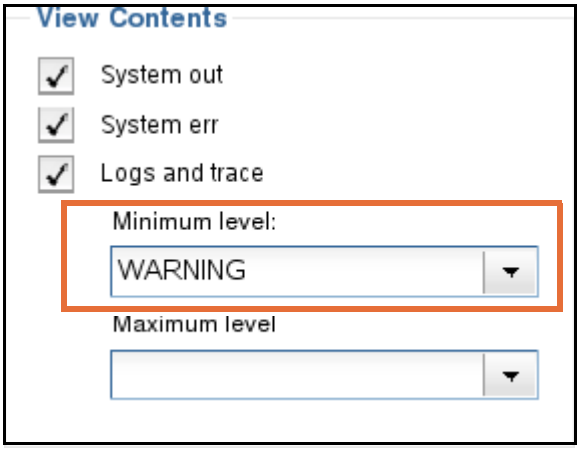
__ a. In the Log Viewer panel, click the **Next Page** and the **Last Page** toolbar buttons to browse though the log records. Then, click the **First Page** toolbar button.



__ b. Try clicking one or more Message IDs links to see more details about a message.

__ 4. Filter the displayed records by message levels.

__ a. Expand **Content and Filtering Details** and select **WARNING** as the Minimum level in the View Contents section.



__ b. Click **Apply**.

__ c. View the records in the Log Viewer. You notice that the records that are displayed have a minimum level of WARNING. Browse through the messages to see whether there are any at a higher level such as SEVERE. Alternatively, select **SEVERE** as the Minimum level in the View Contents section, and click **Apply**.

__ d. Clear the **Minimum** and **Maximum** level windows. Click **Apply** to see all the records again.

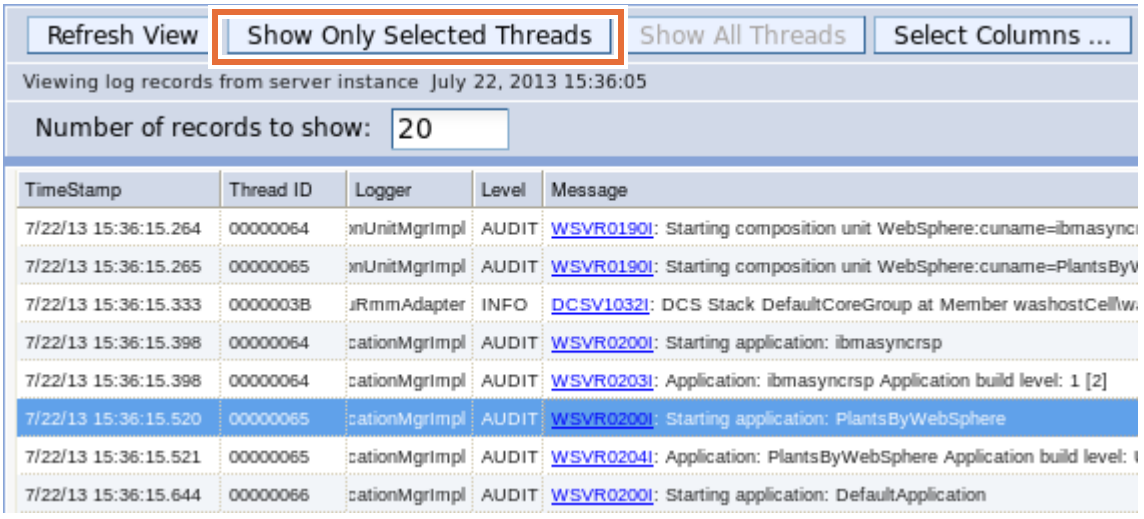
__ 5. Display all records that are associated with a specific thread.

__ a. Browse through the records and look for any message of interest at level WARNING, AUDIT, or SEVERE. For example, **"Starting application: PlantsByWebSphere"**. Record the Thread ID for that record: _____

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Highlight any record with the same Thread ID and click **Show Only Selected Threads**.



__ c. Browse through the resulting records, and you can see that only the messages from the selected Thread ID are displayed. Also, notice that those records are displayed in nearly chronological order (the order in which the server emitted them).

__ d. After viewing the records for the selected thread, click **Show All Threads**.

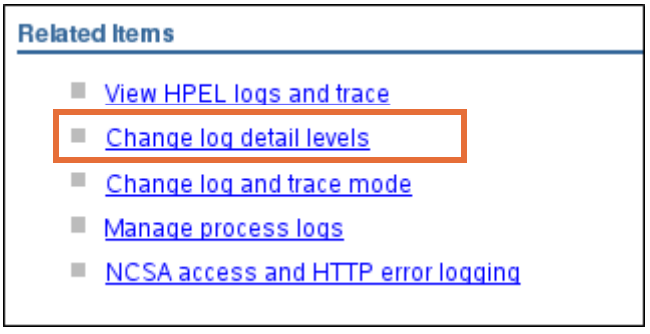
Section 4: Enable tracing for an application server and view trace data by using the Log Viewer

In this section, you enable tracing on the session management components of server1. Use the PlantsByWebSphere Application to generate trace data, and view the trace data in the Log Viewer.

__ 1. Configure the diagnostic trace for the session management components of server1.

__ a. In the administrative console, click **Troubleshooting > Logs and trace > server1**.

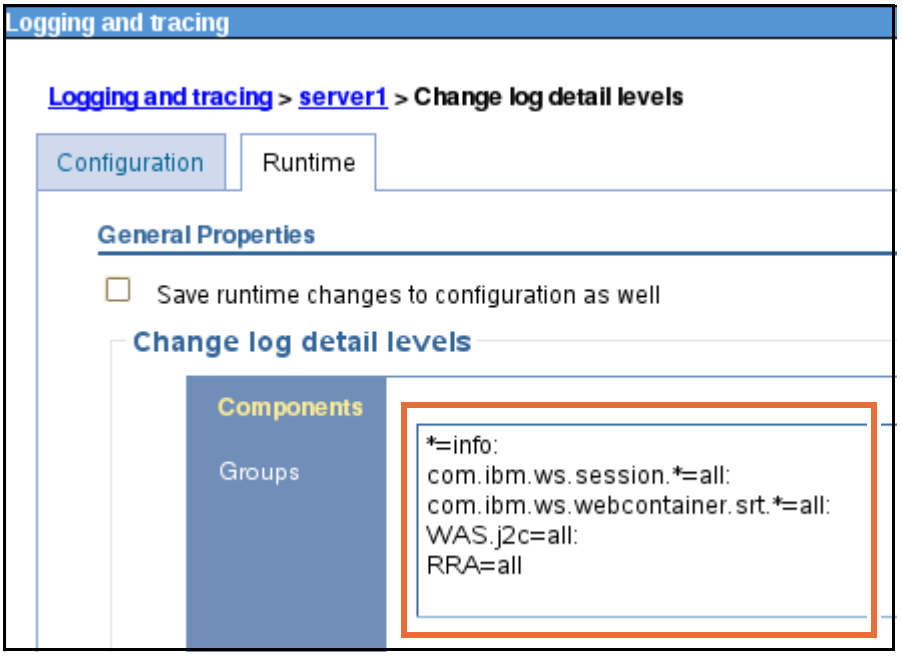
__ b. Click **Change log detail levels**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select the **Runtime** tab, and enter the following trace strings. These strings are for the components and trace levels that are suggested in the IBM Support MustGather documentation for session management problems.

```
com.ibm.ws.session.*=all:
com.ibm.ws.webcontainer.srt.*=all:
WAS.j2c=all:
RRA=all
```



__ d. Remember to add a colon (:) after the existing trace string `*=info:`

__ e. Click **Apply**.

__ 2. Access the PlantsByWebSphere application and generate some trace data.

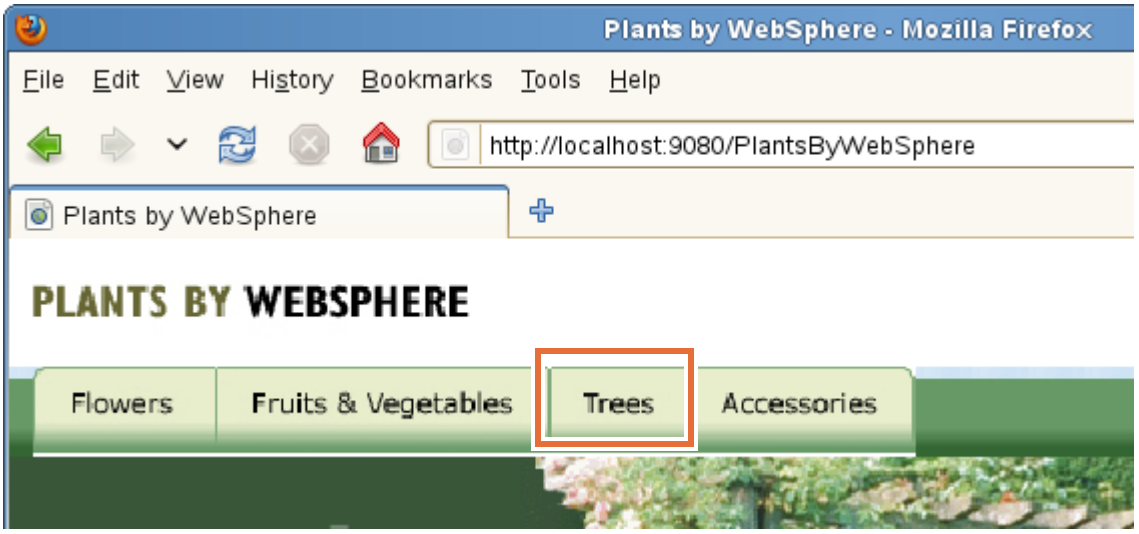
__ a. Start a new browser and enter the web address:

`http://washost:9080/PlantsByWebSphere`

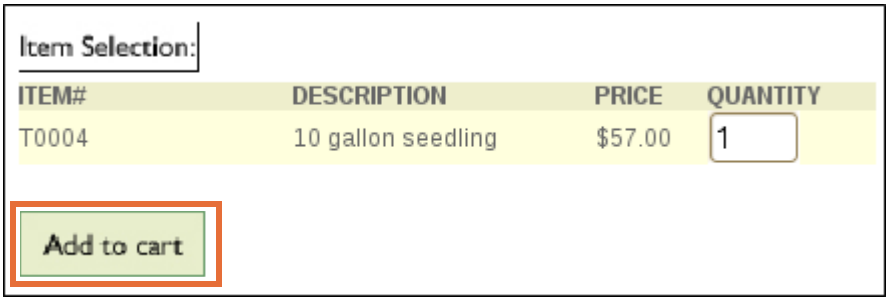
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Click the **Trees** tab on the Welcome page.



__ c. Select any tree that you like, and click **Add to cart**.



__ d. This activity is enough to generate some interesting trace data. Close the browser.

__ 3. Use the Log Viewer in the administrative console to examine the trace data.

__ a. Click **Troubleshooting > Logs and trace > server1 > View HPEL logs and trace**.

__ b. Expand the **Contents and Filtering Details** tree.

Information

In the View Contents section, you can select **System out**, **System err**, and **Logs and trace**.

Selecting **System out** shows records that were sent to the System Out stream by using an API such as `System.out.println(...)`

Selecting **System err** shows records that were sent to the System Error stream by using an API such as `System.err.println(...)`

Selecting **Logs and trace** specifies that log and trace records are included in the log view. Log and trace entries can be further specified to include a minimum or maximum level.

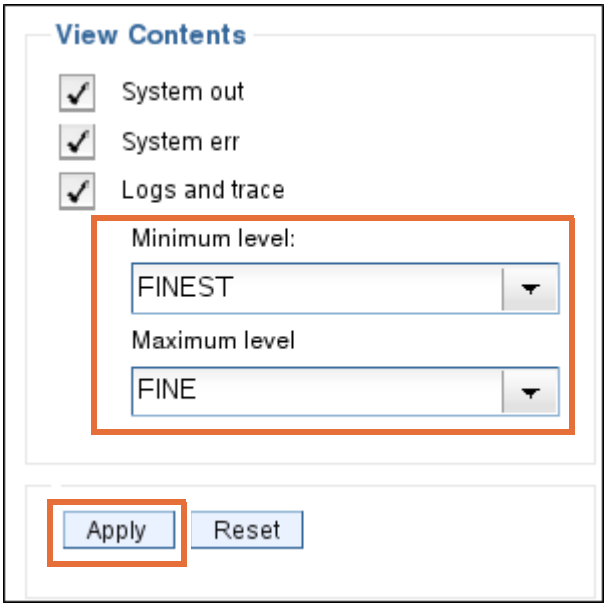
Examples of log and trace filtering:

- The result of selecting logs and trace and clearing Minimum level and Maximum level fields is that the log view displays records with any log or trace level (default).

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

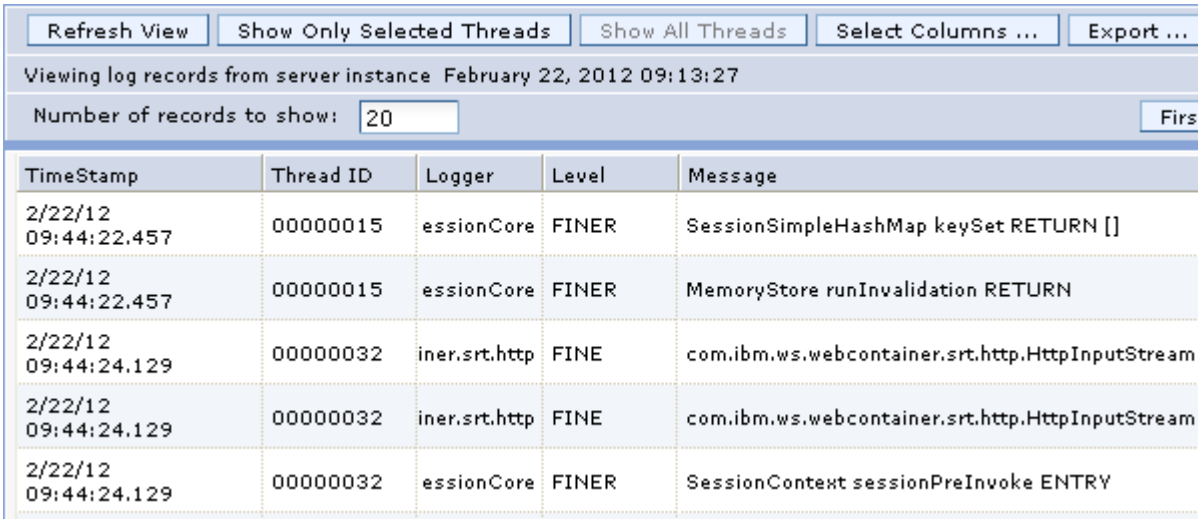
- The result of selecting logs and trace and setting Minimum level to WARNING is that the log view displays log records with levels WARNING, FATAL, or SEVERE.
- The result of selecting logs and trace and setting Maximum level to FINE is that the log view displays trace records with levels FINE, FINER, or FINEST.
- The result of selecting logs and trace and setting Minimum level to DETAIL and Maximum level to AUDIT is that the log view displays log records with levels DETAIL, CONFIG, INFO, or AUDIT.

__ c. In the View Contents section, select **FINEST** as the Minimum level and **FINE** as the Maximum level.



__ d. Click **Apply**.

__ e. Click **Refresh View**.



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

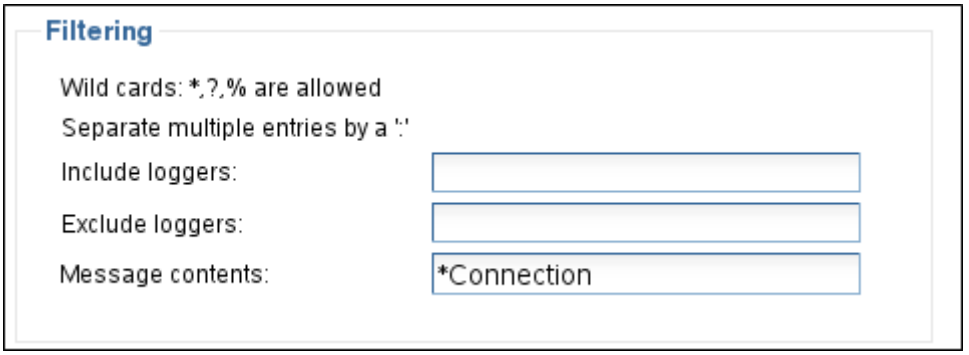
__ f. Use the **Next Page** and **Previous Page** buttons to navigate through the trace data. The level of all records is either FINE, FINER, or FINEST.

__ 4. Use the features of the Log Viewer to explore the trace data. Here are a few suggestions.

__ a. Set both Minimum level and Maximum level to **FINEST**. This setting shows you only the records at the FINEST level, if there are any.

__ b. Select (highlight) any Thread ID of interest and click **Show Only Selected Threads**. Observe the number of different loggers that stream messages in that thread.

__ c. Try filtering on message content. Look for key words among the message details, such as `*getConnection*` or `*JSESSIONID*`. Use wildcards. Remember to click **Apply** and **Refresh View**. Remove the ***Connection** filter when done.



Section 5: Enable cross-component trace (XCT)

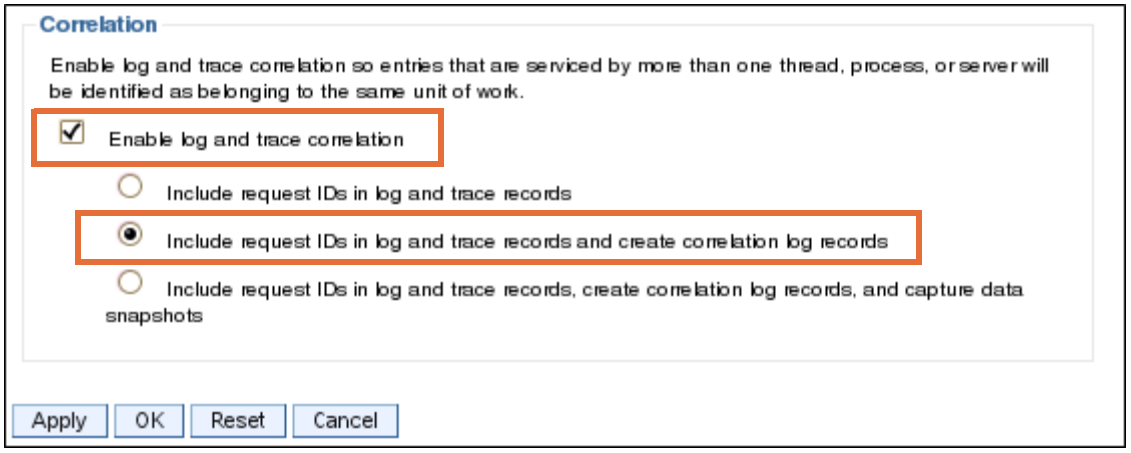
In this section, you learn how to enable cross-component trace (XCT) for an application server. You also examine the request IDs and other data that XCT provides in the server logs.

__ 1. Enable XCT for server1.

__ a. In the administrative console, click **Troubleshooting > Logs and trace > server1 > Change log detail levels**.

__ b. On the **Runtime** tab, scroll down to the Correlation section.

__ c. Check the box for **Enable log and trace correlation** and select **Include request IDs in log and trace records and create correlation log records**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Click **OK**.

Information

Include request IDs and trace records and create correlation log records:

This setting enables XCT to include request IDs in log and trace files when you want to see which log and trace entries, in all threads and application server processes, are related to the same request. Request IDs are recorded only when using HPEL log and trace mode and can be seen or used for filtering by using the `logViewer` command.

In addition, XCT creates correlation log records when you want to log how requests branch between threads and processes, and see extra information about each request.

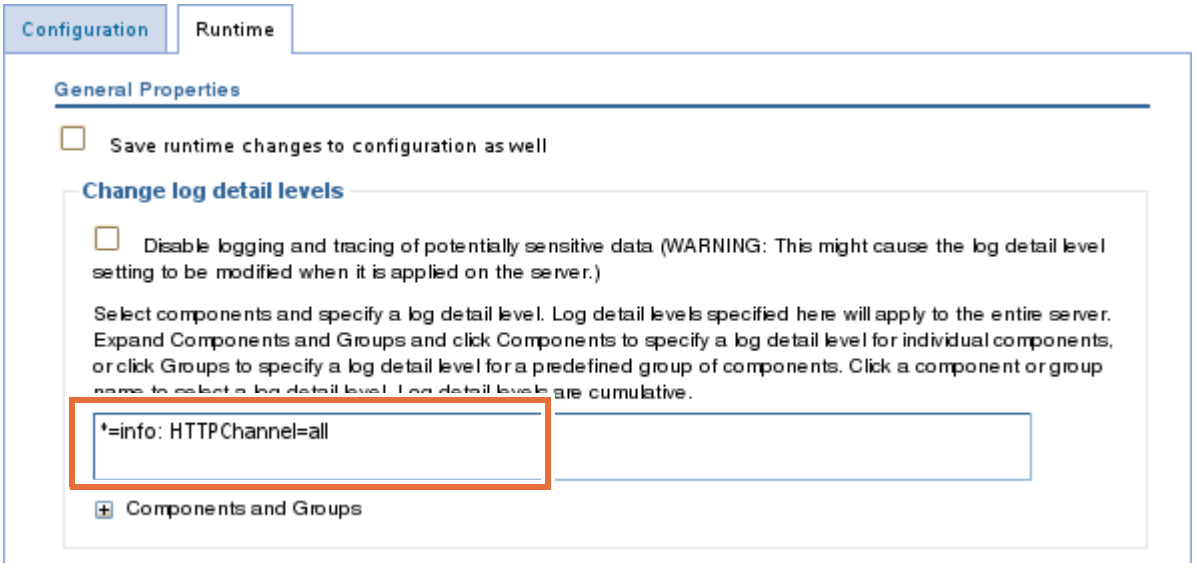
Warning: Enabling XCT to create correlation log records might have a significant performance impact on your system, so is best suited to test and development environments.

__ 2. Change the trace specification to gather data for HTTP requests.

__ a. Click **Troubleshooting > Logs and trace > server1 > Change log detail levels**.

__ b. Select the **Runtime** tab, and add the following trace string:

`*=info: HTTPChannel=all`



__ c. Clear the previous setting and click **OK**.

__ 3. Generate an HTTP request and examine the XCT data by using the GUI Log Viewer.

__ a. Start a new browser and enter the web address:

`http://washost:9080/PlantsByWebSphere`

Accessing the Plants By WebSphere Welcome page generates several HTTP requests.

__ b. From the administrative console, click **Troubleshooting > Logs and trace > server1 > View HPEL Logs and Trace**.

© Copyright IBM Corp. 2013

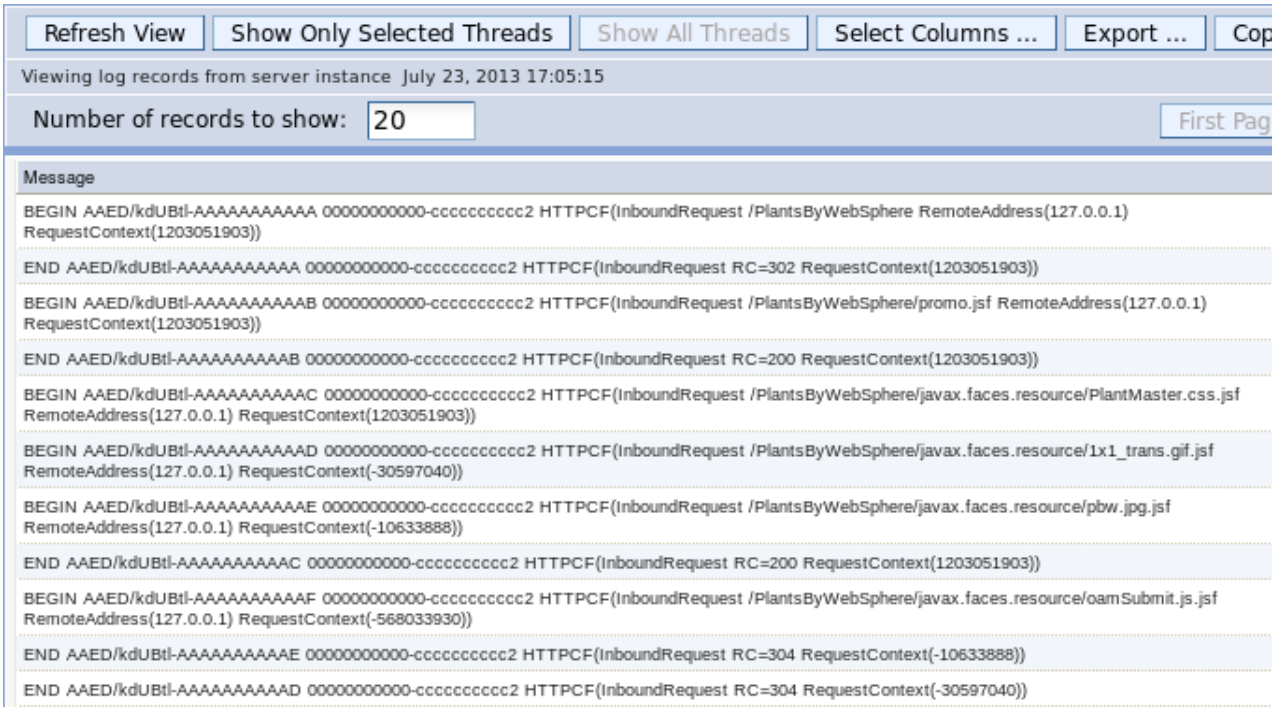
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Expand the **Contents and Filtering Details** tree.

__ d. Under the Filtering section, enter `com.ibm.websphere.XCT` in the Include loggers field.

__ e. Click **Apply**. Notice that the log records from the `com.ibm.websphere.XCT` logger are displayed.

__ f. Click **Select columns**, and select only **Message**. Clear the other columns and click **OK** to see the view that is shown in the screen capture.



Information

When an HTTP request arrives, the server does an XCT BEGIN that indicates that the request started processing. The entry in the logs shows the following information:

- Parent XCT ID
- Current XCT ID
- Type of request (InboundRequest or OutboundRequest)
- URI of request RequestContext object
- ID from HTTPChannel

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

- RemoteAddress from the connection that the request is originated from

When the request completes processing, the server does an XCT END; this message marks the request as finished. An XCT END for an HTTP request shows the following attributes:

- Parent XCT ID
- Current XCT ID
- The type of request, which includes InboundRequest and OutboundRequest.
- Return code (RC) of the response
- HTTP Channel RequestContext object ID

__ 4. Use the command line Log Viewer to examine the XCT data

__ a. Navigate to: `<profiles_root>/profile1/bin`

__ b. Find the instance ID of the running server by entering the following command:

```
./logViewer.sh -listInstances
```

Enter the last instance ID here: _____

__ c. Enter the following command and redirect the output.

```
./logViewer.sh -includeLoggers "com.ibm.websphere.XCT" -format advanced
-instance <your_Instance_ID> > /usr/Student/PDTools/xct_data.txt
```

__ d. When this command completes, the log records are written to the text file by using the advanced format. Open the `xct_data.txt` file with an editor such as gedit to explore its contents.

```
[time_stamp] 00000059  I UOW= source=com.ibm.websphere.XCT org=null prod=null component=null
thread=[WebContainer : 0] requestID=[AAABvgxR9nQ-AAAAAAAAAAAA]
        BEGIN AAABvgxR9nQ-AAAAAAAAAAAA 00000000000-cccccccccc2 HTTPCF(InboundRequest
/PlantsByWebSphere RemoteAddress(127.0.0.1) RequestContext(276984898))

[time_stamp] 00000059  I UOW= source=com.ibm.websphere.XCT org=null prod=null component=null
thread=[WebContainer : 0] requestID=[AAABvgxR9nQ-AAAAAAAAAAAA]
        END   AAABvgxR9nQ-AAAAAAAAAAAA 00000000000-cccccccccc2 HTTPCF(InboundRequest
RC=302 RequestContext(276984898))
. . .
        BEGIN AAABvgxR9nQ-AAAAAAAAAAAE 00000000000-cccccccccc2 HTTPCF(InboundRequest
/PlantsByWebSphere/javax.faces.resource/oamSubmit.js.jsf RemoteAddress(127.0.0.1)
RequestContext(1918170885))

        END   AAABvgxR9nQ-AAAAAAAAAAAE 00000000000-cccccccccc2 HTTPCF(InboundRequest
RC=200 RequestContext(1918170885))
```

__ e. Notice that the advanced format shows the unit of work (UOW), in this case the XCT logger and the requestID.

__ f. Close the `xct_data.txt` file when you are finished examining it.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

IBM Cross Component Trace Log Viewer

Available in the IBM Support Assistant, IBM WebSphere Cross Component Trace Log Viewer provides enhanced log file views for logs that are augmented with Cross Component Trace correlation log records. Logs can be displayed in flat or hierarchical layouts, and multiple logs can be loaded and viewed simultaneously with log entries related to each request conveniently grouped.

__ 5. Clear the HTTP trace strings and disable XCT.

__ a. Click **Troubleshooting > Logs and trace > server1 > Change log detail levels**.

__ b. On the Runtime tab, remove the trace string: `HTTPChannel=all`

__ c. Scroll down to the Correlation section and clear the box for **Enable log and trace correlation**.

__ d. Click **OK**.

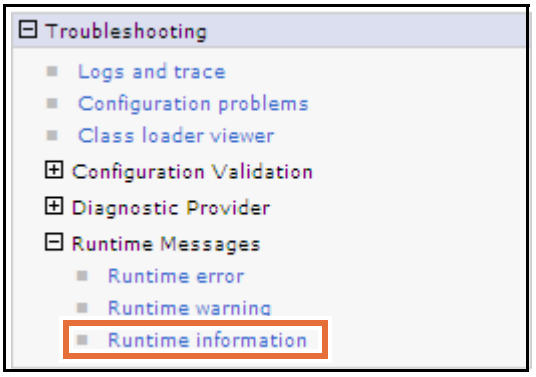
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 6: Examine runtime messages

The Troubleshooting area of the administrative console displays messages about runtime events and configuration problems. This area automatically refreshes, and you can view either the runtime messages or configuration problem totals.

__ 1. In the administrative console navigation tree, expand **Troubleshooting**.

__ 2. Expand **Runtime Messages**. You see entries for **Runtime Error**, **Runtime Warning**, and **Runtime Information**.



__ 3. Click **Runtime Information** to view all of the messages. The Runtime events option is disabled by default; the event level is “None”, so you do not see any messages. Enable all runtime events.

   __ a. Select **Info** from the menu, and click **Apply**.

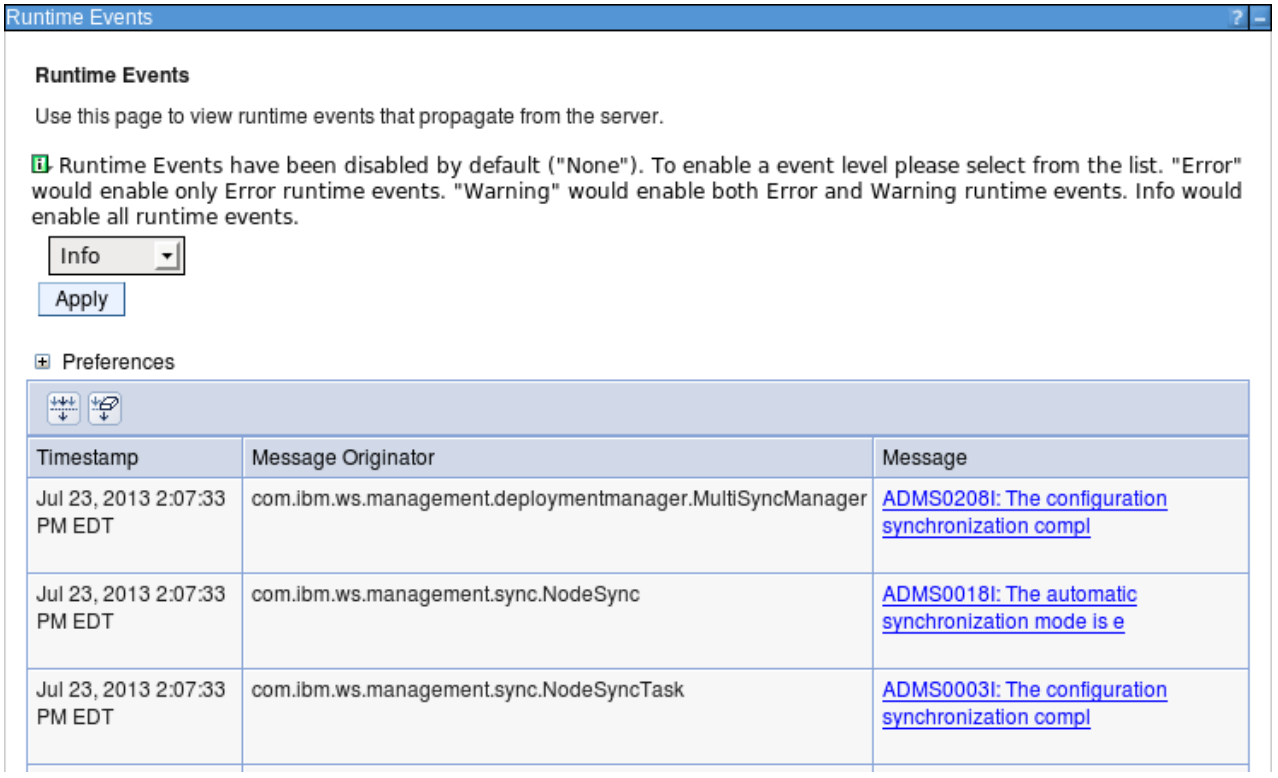
   __ b. **Save** the changes to the master configuration

   __ c. Wait for the node to synchronize, and click **OK**. Several messages are now shown.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. In the **Message** column, click one of the messages (if there are any listed) to see the message details.



Information

Configuration validation

You can configure the validation policy by clicking **Troubleshooting > Configuration Problems**. You can set the validation policy to any of the following options: None, Low, Medium, High, or Maximum. Low is the default policy. When troubleshooting configuration problems, you must set the policy to High or Maximum.

Selecting **High: Validate extracted, parent, and sibling documents** turns on validation for extracted documents and their parent documents, and turns on validation for the sibling documents of the documents that are extracted. For example, if High validation is selected, and if the `server.xml` document is extracted, validation is done on the three documents: `server.xml`, `node.xml`, and `cell.xml`. Validation is also done on the two sibling documents, `variables.xml` and `resources.xml`, within the server1 directory.

See the online help for information about the other policies.

Section 7: Examine FFDC logs

The first-failure data capture (FFDC) log files save information that is generated from a processing failure (for example, a Java exception). Captured data is saved in log files for use in analysis. These files include an index file that references all of the exceptions that FFDC logs, and an exception file

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

for each exception type from each probe. These files are deleted after a maximum number of days. FFDC is enabled by default since capturing FFDC data does not affect performance.

__ 1. Examine the FFDC logs for profile01.

__ a. Use Windows Explorer to navigate to `<profiles_root>/profiles/profile1/logs/ffdc`

__ b. Typically you see dozens of files in the `ffdc` directory. These files include:

- `nodeagent.<Thread_ID>.<Timestamp>.<Seq_No>.txt` (several)
- `nodeagent_exception.log`
- `<app_server_name>.<Thread_ID>.<Timestamp>.<Seq_No>.txt` (several)
- `<app_server_name>_exception.log` (one per application server)

The `SystemOut.log` for the node agent and the application servers contains messages that the FfdcProvider writes for each failure incident. These messages point to the relevant text file in the `ffdc` directory, which contains the exception and the stack trace (call stack).

__ 2. Examine a particular FFDC incident.

__ a. Use Notepad to open one of the older server exception logs with a name like the `server1_exception.log.<Seq_No>.txt` file. Each row has an index and a count number. The count is the number of occurrences of the exception and is useful for seeing whether there is an unusual number of any particular exception.

__ b. Select any exception of interest and record its text file name:

_____

__ c. Locate the text file in the `ffdc` directory and open it with Notepad. In this file, you see the actual Java exception that was thrown and the stack trace.

__ d. After you are done examining the text file, close it.

Section 8: Work with the Classloader viewer

When enabled for an application server, the class loader viewer service allows you to use the administrative console Class loader viewer to examine class loaders and the classes that each class loader loads.

__ 1. Enable the class loader viewer service

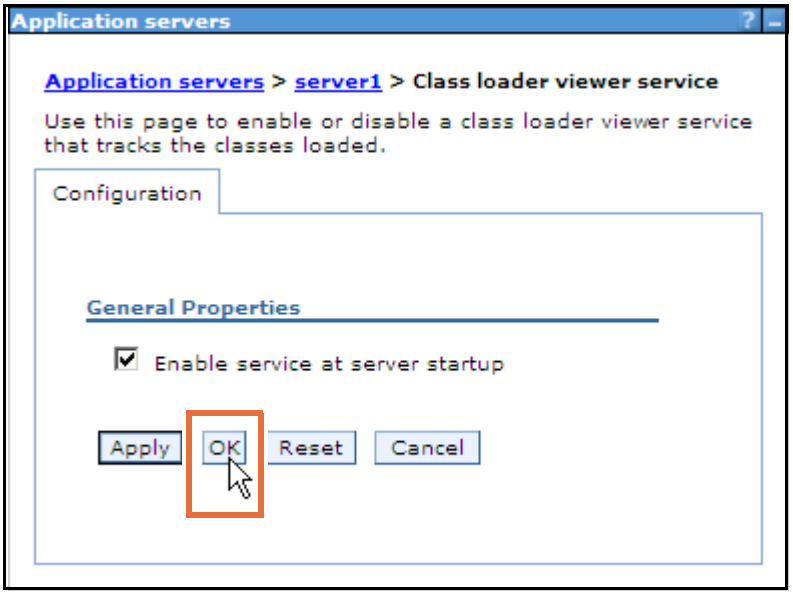
__ a. Select **Servers > Server Types > WebSphere application servers > server1**.

__ b. Under **Additional Properties**, click **Class loader viewer service**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select the **Enable service at server startup** check box and then click **OK**.



__ d. **Save** the changes to the master configuration.

__ e. Wait for the node to synchronize and click **OK**.

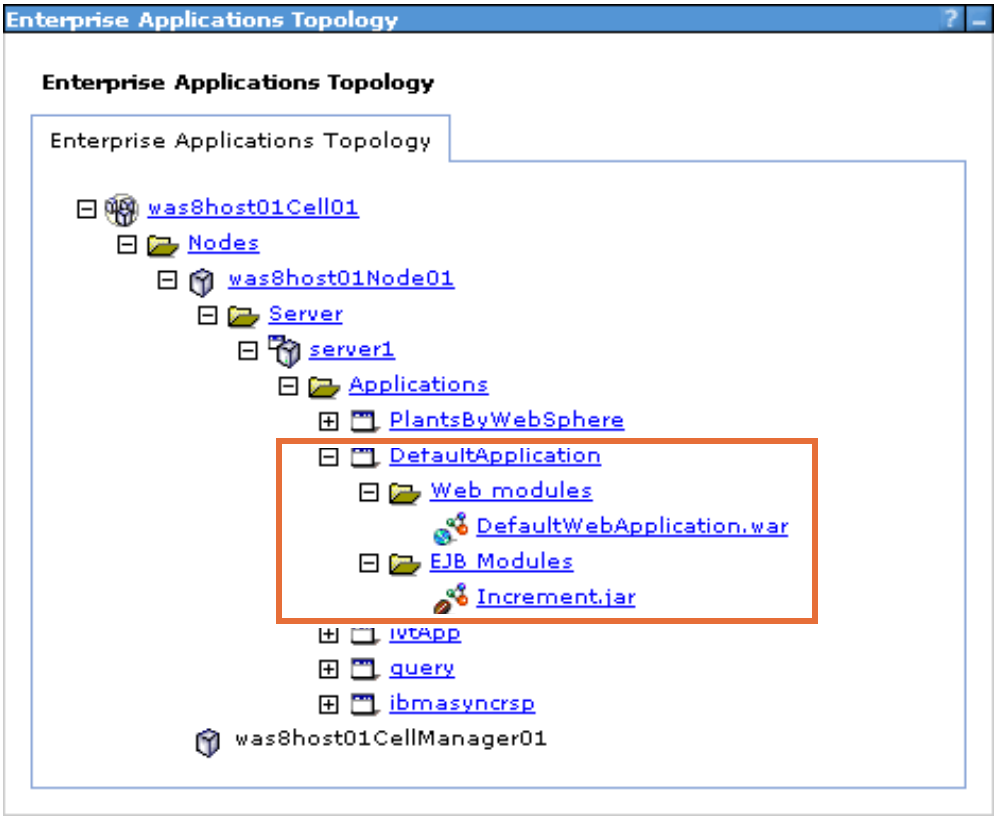
__ f. Restart **server1** from the administrative console.

__ 2. Examine class loaders.

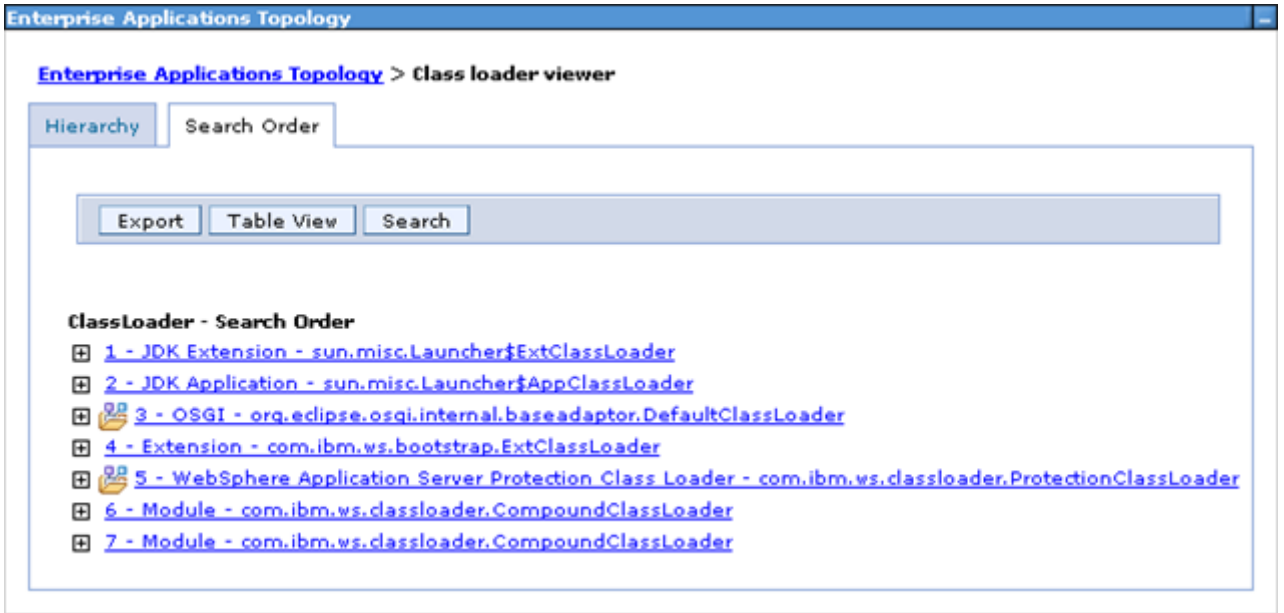
__ a. Select **Troubleshooting > Class loader viewer** to access the Enterprise Applications Topology page. Use this page to see where modules are contained in a topology of enterprise applications. Knowing where a module is contained helps you to determine which class loader loaded a module and to diagnose problems with class loaders.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the **Enterprise Applications Topology** page, expand the tree to show the entries under **DefaultApplication**.



__ c. To view a class loader, click any module link. Click the DefaultApplication web module, **DefaultWebApplication.war**.



__ d. Select the **Search Order** tab. You can use this page to examine the class loaders visible to a web module (`.war` file) in an installed enterprise application. This page helps you to determine which class loaders loaded files of a module.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. You can expand any of the class loaders to see which classes they load, and where the classes are located. Try expanding the **Module** class loaders.

__ f. If you click the **Table View,** you can view information about class loaders visible to the module in a table format. Such information includes: delegation, class path, and class names.

Section 9: Work with namespace tools

Clients of Java EE applications use naming to obtain references to objects related to those applications, such as enterprise bean (EJB) homes.

These objects are bound into a mostly hierarchical structure, called a namespace. In this structure, all non-leaf objects are called contexts. Leaf objects can be contexts and other types of objects. Naming operations, such as lookups and binds, are performed on contexts. All naming operations begin with obtaining an initial context. You can view the initial context as a starting point in the namespace.

__ 1. Use the dumpNameSpace tool to dump namespaces from different root contexts and different name servers.

You can use the dumpNameSpace tool to dump the contents of a namespace that is accessed through a name server. The dumpNameSpace tool is based on Java Naming and Directory Interface (JNDI). When you run the dumpNameSpace tool, the naming service must be active.

__ a. In a terminal window change directory to:

`<profiles_root>/profiles/profile1/bin`

__ b. To dump the namespace from the cell root by using the name service of server1, enter the following command:

```
./dumpNameSpace.sh -root cell -port 9809 -user wasadmin -password
web1sphere | more
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Examine the memory dump.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./dumpNameSpace.sh -root cell -port 9809 -user wasadmin -password web
1sphere

Getting the initial context
Getting the starting context

================================================================================
Name Space Dump
   Context factory: com.ibm.websphere.naming.WsnInitialContextFactory
   Provider URL: corbaloc:iiop:localhost:9809
   Requested root context: cell
   Starting context: (top)=washostCell
   Formatting rules: jndi
   Time of dump: Tue Jul 23 14:51:18 EDT 2013
================================================================================



================================================================================
Beginning of Name Space Dump
================================================================================

    1 (top)
    2 (top)/applications                                  javax.naming.Context
    3 (top)/applications/ManagementEJB                    javax.naming.Context
    4 (top)/applications/ManagementEJB/mejb               javax.naming.Context
    5 (top)/applications/ManagementEJB/mejb/Management   javax.management.j2ee.Management
Home
    6 (top)/applications/ManagementEJB/mejb/Management!javax.management.j2ee.ManagementHo
me
    6                                                     javax.management.j2ee.Management
Home
    7 (top)/applications/cell                             javax.naming.Context
    7    Linked to context: washostCell
    8 (top)/applications/com.ibm.ws.AppNameSpaces         javax.naming.Context
```

__ d. Take time to scroll down and examine the Name Space Dump. You see JNDI names for JDBC, JTA, and other application resources.

__ e. Try dumping the namespace from the server1 context root for comparison. Enter the command:

```
./dumpNameSpace.sh -root server -port 2809 -user wasadmin -password
web1sphere
```

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Examine the memory dump.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./dumpNameSpace.sh -root server -port 2809 -user wasadmin -password we
b1sphere

Getting the initial context
Getting the starting context

================================================================================
Name Space Dump
   Context factory: com.ibm.websphere.naming.WsnInitialContextFactory
   Provider URL: corbaloc:iiop:localhost:2809
   Requested root context: server
   Starting context: (top)=washostCell/nodes/washostNode01/servers/server1
   Formatting rules: jndi
   Time of dump: Tue Jul 23 15:26:20 EDT 2013
================================================================================



================================================================================
Beginning of Name Space Dump
================================================================================

    1 (top)
    2 (top)/wm                                          javax.naming.Context
    3 (top)/wm/default                                  com.ibm.websphere.asynchbeans.Wor
kManager
    4 (top)/wm/ard                                      com.ibm.websphere.asynchbeans.Wor
kManager
    5 (top)/UserRegistry                                com.ibm.websphere.security._UserR
egistry_Stub
    6 (top)/jdbc                                        javax.naming.Context
    7 (top)/jdbc/DefaultEJBTimerDataSource              javax.resource.cci.ConnectionFact
ory
    8 (top)/jdbc/PlantsByWebSphereDataSource            javax.resource.cci.ConnectionFact
ory
    9 (top)/cell                                        javax.naming.Context
    9    Linked to context: washostCell
```

__ g. Notice that dumping from the server context enables you to see what resources (for example, data sources, EJBs, connection factories) are visible to applications that are running in the server. Scroll down and examine the whole memory dump.

Information

You cannot use the **dumpNameSpace** tool to dump a `java:` or `local:` namespace because the dumpNameSpace tool cannot access those namespaces.

The `java:` namespace of a Java Platform, Enterprise Edition (Java EE) application is accessible only by that application. You can invoke a NameServer MBean to dump the java namespace for any Java EE application that is running in the same server process.

The `local:` namespace contains references to enterprise beans with local interfaces. There is only one `local` namespace in a server process. You can invoke the NameServer MBean associated with that server process to dump the local namespace.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Use the scripting tool to invoke the NameServer MBean that runs in the application's server process to generate memory dumps of java, local, or server namespaces.

__ 2. Use the wsadmin scripting tool to invoke the NameServer MBean.

In a command prompt change directory to:
<profiles_root>/profiles/profile1/bin

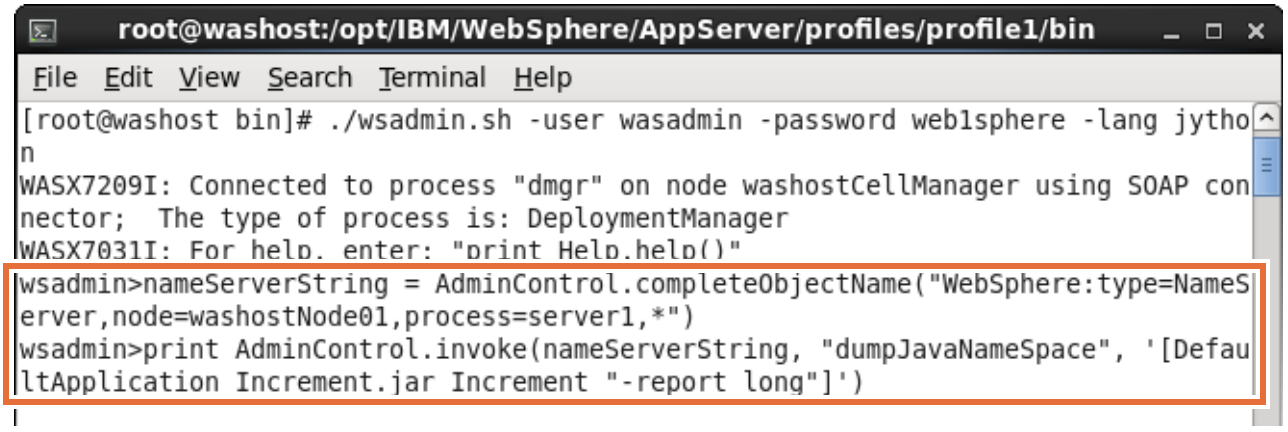
__ h. Invoke wsadmin by entering the following command:

```
./wsadmin.sh -user wasadmin -password web1sphere -lang jython
```

__ i. Since this time is the first time that you are starting wsadmin, you see all the jython classes loading.

__ j. Assume that you want to dump the `java:` namespace of an application component that is running in server1 on node `washostNode01` of the cell `washostCell`. The application name is `DefaultApplication` in module `Increment.jar`, and the component name is `Increment`. Enter the following script commands to generate a long format dump of the application's `java:` namespace. If you like, you can copy and paste these commands from the file `/usr/Student/PDTools/namespacedump_cmds.txt`.

```
nameServerString =
AdminControl.completeObjectName("WebSphere:type=NameServer,node=washostNo
de01,process=server1,*")
print AdminControl.invoke(nameServerString, "dumpJavaNameSpace",
'[DefaultApplication Increment.jar Increment "-report long"]')
```



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ k. Examine the memory dump.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
================================================================================
"Name Space Dump"
"   Starting context: (top)=java:"
"   Time of dump: Tue Jul 23 15:44:59 EDT 2013"
================================================================================
" "
" "
================================================================================
"Beginning of Name Space Dump"
================================================================================
" "
" "
"    1 (top)"
"    1    Bound Java type: "
"    1    Local Java type: com.ibm.ws.naming.java.javaURLContextRoot"
"    1    Corba binding type: org.omg.CosNaming.BindingType.ncontext"
"    1    Context unique ID: ROOT CONTEXT"
"    1    String representation: com.ibm.ws.naming.java.javaURLContextRoot@a796f8b3[super=
com.ibm.ws.naming.java.javaURLContextImpl@a796f8b3[super=com.ibm.ws.naming.urlbase.UrlCont
extImpl(java:)@a796f8b3[_schemeId=java, _primaryLeafName=null, _strCtxID=ROOT CONTEXT, _ns
=com.ibm.ws.naming.ipbase.NameSpace@a4fe36a8[_nameSpaceName=java:comp(DefaultApplication,
Increment, Increment), _nameSpaceID=16, _nameSpaceType=1, _contextsTable=java.util.HashMap
@1e6b288a[size=5], _bindingsTable=java.util.HashMap@dccd50f5[size=20]]], _javaNameSpaceSco
pe=COMPONENT SCOPE, _jns=javaNameSpaceImpl:{_appName=DefaultApplication,_moduleName=Increm
ent,_componentName=Increment,_bootstrapAddress=rir:,_clientMode=SERVER_DEPLOYED,_component
NameSpaceLocation=LOCAL,_otherModuleNames=[],_applicationsRootProviderURL=corbaname:rir:/N
ameServiceApplicationsRoot#,_applicationRootProviderURL=corbaname:rir:/NameServiceApplicat
ionsRoot#com\.ibm\.ws\.AppNameSpaces/DefaultApplication/root,_moduleRootProviderURL=null,_
componentRootProviderURL=null,_ns=com.ibm.ws.naming.ipbase.NameSpace@a4fe36a8[_nameSpaceNa
me=java:comp(DefaultApplication, Increment, Increment), _nameSpaceID=16, _nameSpaceType=1,
 _contextsTable=java.util.HashMap@1e6b288a[size=5], _bindingsTable=java.util.HashMap@dccd5
0f5[size=20]]}], _caching=true, _rootCache=com.ibm.ws.naming.java.JavaNameSpaceCache@4c4c5
d7b]"
" "
"    2 (top)/comp"
"    2    Bound Java type: com.ibm.ws.naming.java.javaURLContextImpl"
"    2    Local Java type: com.ibm.ws.naming.java.javaURLContextImpl"
"    2    Corba binding type: org.omg.CosNaming.BindingType.ncontext"
"    2    Context unique ID: cf2307e-0140-fefa-0abf-7f000001aa77"
"    2    String representation: com.ibm.ws.naming.java.javaURLContextImpl@9abc8959[super=
com.ibm.ws.naming.urlbase.UrlContextImpl(java:comp)@9abc8959[ schemeId=java,  primaryLeafN
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ l. If you want to dump the `local:` namespace for the server `server1` on node `washostNode01` of cell `washostCell`, enter the following script commands to generate a long format dump of that server's local namespace. This command assumes that you already ran the command to set `nameServerString` that is given in the previous step.

```
print AdminControl.invoke(nameServerString, "dumpLocalNameSpace",
'["-report long"]')
```

__ m. Examine the memory dump.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
wsadmin>print AdminControl.invoke(nameServerString, "dumpLocalNameSpace", '["-re
port long"]')

================================================================================
Name Space Dump
   Starting context: (top)=local:
   Time of dump: Wed Jul 24 10:02:38 EDT 2013
================================================================================



================================================================================
Beginning of Name Space Dump
================================================================================



    1 (top)
    1    Bound Java type:
    1    Local Java type: com.ibm.ws.naming.urlns.genericURLContextRoot
    1    Corba binding type: org.omg.CosNaming.BindingType.ncontext
    1    Context unique ID: ROOT CONTEXT
    1    String representation: com.ibm.ws.naming.urlbase.UrlContextImpl(local:)
@b68b2550[_schemeId=local, _primaryLeafName=null, _strCtxID=ROOT CONTEXT, _ns=co
m.ibm.ws.naming.ipbase.NameSpace@108635a7[_nameSpaceName=null, _nameSpaceID=4, _
nameSpaceType=1, _contextsTable=java.util.HashMap@0[size=0], _bindingsTable=java
.util.HashMap@0[size=0]]]
```

Note

As you can see, namespace dump output is sent to the console. However, it is also written to the file `DumpNameSpace.log` in the server's log directory if you need to analyze the namespace further.

End of exercise

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In this exercise, you learned how to locate and view log files for the WebSphere Application Server. You learned how to enable HPEL logging and XCT for an application server. You also learned how to enable tracing for an application server by using the administrative console. You examined other sources of diagnostic data about class loaders by using the Class Loader Viewer, and compared namespaces by using the dumpNameSpace tool and name server MBean.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 3. Introduction to configuring garbage collection policies

What this exercise is about

This exercise demonstrates how to configure different GC policies for an application server. You configure and examine two different GC policies, optthruput and gencon.

What you should be able to do

At the end of this exercise, you should be able to:

- Enable verbose GC for an application server
- Manually view verbose GC logs
- Configure the optthruput GC policy
- Configure the gencon GC policy
- Use IBM Monitoring and Diagnostic Tools for Java – Garbage Collection and Memory Visualizer (GCMV) to analyze verbose GC data
- Configure and use the Java Health Center

Introduction

When the JVM cannot allocate an object from the heap because of lack of contiguous space, a memory allocation fault occurs, and the Garbage Collector is called. The first task of the Garbage Collector is to collect all the garbage that is in the heap. This process starts when any thread calls the Garbage Collector either indirectly as a result of allocation failure, or directly by a specific call to `System.gc()`.

In this exercise, you configure and explore two different GC modes or policies. The GC policy, optthruput, attempts to optimize throughput for your applications. The default GC policy for beginning in WebSphere Application Server V8, gencon, attempts to provide both optimized throughput and response times. In addition, the gencon GC policy virtually eliminates heap fragmentation by using a partitioned heap.

Requirements

In addition to the tools available with WebSphere Application Server, you use tools that are available through the IBM Support Assistant. Specifically, you

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

use the Garbage Collection and Memory Visualizer tool (GCMV), and the Java Health Center (JHC).

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Enable verbose GC for an application server

You use the administrative console to enable verbose GC for server1.

__ 1. Ensure that the deployment manager is started.

__ a. From a terminal window, change directory to:

`<profiles_root>/Dmgr/bin`

__ b. If dmgr is stopped, start it with the command: `./startManager.sh`

__ 2. Ensure that the node agent is started and server1 is stopped.

__ a. From a terminal window, change directory to:

`<profiles_root>/profile1/bin`

__ b. If the node agent is stopped, start it with command: `./startNode.sh`

__ c. If server1 is started, stop it with command:

```
./stopServer.sh server1 -user wasadmin -password web1sphere
```

__ 3. Use the administrative console to enable verbose GC for server1.

__ a. Log on to the administrative console as follows:

- User ID: `wasadmin`
- Password: `web1sphere`

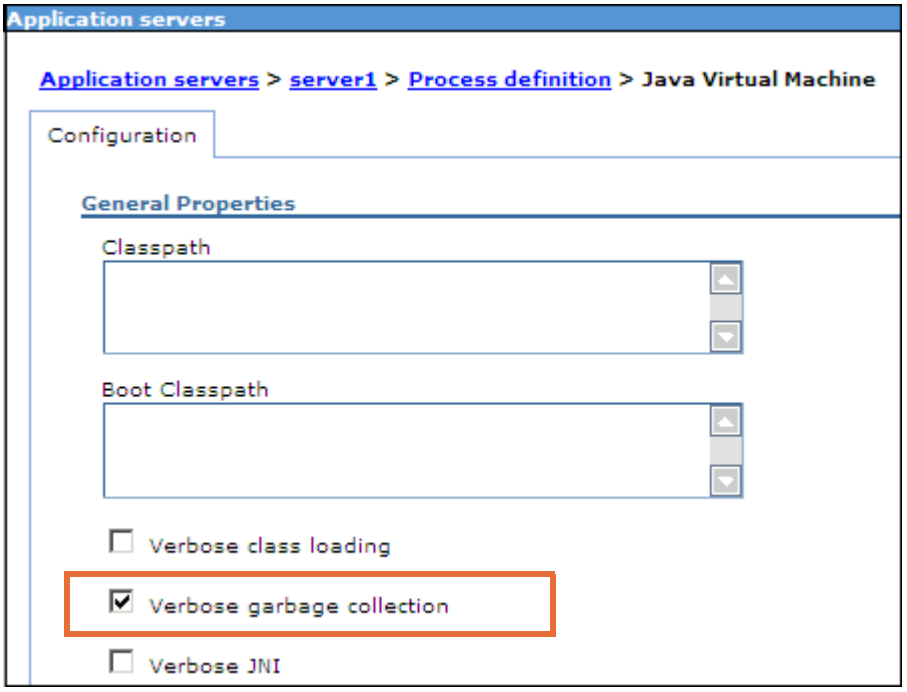
__ b. Click **Servers > Server Types > WebSphere application servers > server1**.

__ c. On the configuration tab for server1, under the **Server Infrastructure** section, click **Java and process management**.

__ d. Click **Process definition > Java Virtual Machine**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. On the JVM configuration tab for server1 check **Verbose garbage collection**.



__ f. Click **OK**.

__ g. Click the link to **Save** directly to master configuration.

__ h. Wait for the nodes to synchronize and click **OK**.

__ 4. Examine the verbose garbage collection data for server1.

__ a. Use a file system browser to navigate to:

`<profiles_root>/profile1/logs/server1`

__ b. Delete the `native_stderr.log` and `native_stdout.log` files in the directory.

__ c. Start **server1** from the administrative console and wait for it to start successfully.

__ d. In the logs directory for server1, open `native_stderr.log` with a text editor such as gedit.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. At the top of the log, examine the text:

```
<initialized id="1" timestamp="2014-02-03T11:04:52.116">
  <attribute name="gcPolicy" value="-Xgcpolicy:gencon" />
  <attribute name="maxHeapSize" value="0x10000000" />
  <attribute name="initialHeapSize" value="0x3200000" />
  <attribute name="compressedRefs" value="true" />
  <attribute name="compressedRefsDisplacement" value="0x0" />
  <attribute name="compressedRefsShift" value="0x0" />
  <attribute name="pageSize" value="0x1000" />
  <attribute name="pageType" value="not used" />
  <attribute name="requestedPageSize" value="0x1000" />
  <attribute name="requestedPageType" value="not used" />
  <attribute name="gcthreads" value="1" />
  <attribute name="numaNodes" value="0" />
```

__ f. The first tag that is shown in the output is the `<initialized>` tag, which precedes values that include an ID and time stamp. The information that is shown in the `<initialized>` section includes the garbage collection policy, the policy options, and any JVM command-line options that are in effect at the time. Notice that the GC policy is generational concurrent (gencon), which is the default policy. Notice that maximum heap size is 256 MB and initial heap size is 50 MB. These values are the default heap sizes for IBM JVMs.

__ g. Scroll down in the log and you see a number of allocation failures `<af-start...af-end>`.

```
<af-start id="3" totalBytesRequested="16" timestamp="2014-02-03T11:04:53.226"
intervalms="1110.692" />
<cycle-start id="4" type="scavenge" contextid="0" timestamp="2014-02-03T11:04:53.227"
intervalms="1110.739" />
<gc-start id="5" type="scavenge" contextid="4" timestamp="2014-02-03T11:04:53.227">
  <mem-info id="6" free="38957896" total="45875200" percent="84">
    <mem type="nursery" free="0" total="6553600" percent="0" />
    <mem type="tenure" free="38957896" total="39321600" percent="99">
      <mem type="soa" free="36991816" total="37355520" percent="99" />
      <mem type="loa" free="1966080" total="1966080" percent="100" />
    </mem>
    <remembered-set count="5049" />
  </mem-info>
</gc-start>
<allocation-stats totalBytes="6912928" >
  <allocated-bytes non-tlh="1160736" tlh="5752192" />
  <largest-consumer threadName="main" threadId="0000000011EE7100" bytes="6885184" />
</allocation-stats>
<gc-op id="7" type="scavenge" timems="6.984" contextid="4" timestamp="2014-02-03T11:04:53.234">
  <scavenger-info tenureage="10" tiltratio="50" />
  <memory-copied type="nursery" objects="18370" bytes="1062880" bytesdiscarded="520" />
  <finalization candidates="118" enqueued="14" />
  <ownableSynchronizers candidates="16" cleared="0" />
  <references type="soft" candidates="23" cleared="0" enqueued="0" dynamicThreshold="32"
maxThreshold="32"/>
  <references type="weak" candidates="23" cleared="0" enqueued="0" />
</gc-op>
<gc-end id="8" type="scavenge" contextid="4" durationms="7.125"
timestamp="2014-02-03T11:04:53.234">
  <mem-info id="9" free="44370048" total="45875200" percent="96">
    <mem type="nursery" free="5412152" total="6553600" percent="82" />
    <mem type="tenure" free="38957896" total="39321600" percent="99">
      <mem type="soa" free="36991816" total="37355520" percent="99" />
      <mem type="loa" free="1966080" total="1966080" percent="100" />
    </mem>
    <pending-finalizers system="14" default="0" reference="0" classloader="0" />
    <remembered-set count="5049" />
  </mem-info>
</gc-end>
<cycle-end id="10" type="scavenge" contextid="4" timestamp="2014-02-03T11:04:53.234" />
<allocation-satisfied id="11" threadId="0000000011EE7100" bytesRequested="16" />
<af-end id="12" timestamp="2014-02-03T11:04:53.234" />
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

Interpreting the verbose GC data

The **intervalms** provides the amount of time in milliseconds (ms) between allocation failures.

`totalBytesRequested` is the size of the object in bytes that caused the allocation failure.

Within the `<gc-start.../gc-start>` section, you can see the size of the nursery and tenure parts of the heap at the time of the allocation failure. In this example, the nursery has 0 free bytes; therefore it is garbage collected.

The `<gc-op>` output contains subsections that describe operations that are specific to different garbage collection policies, **gencon** in this example. These subsections might change from release to release, when improvements are made to the technology or when new data becomes available.

Within the `<gc-op.../gc-op>` section, are the details of the garbage collection. `type="scavenger"` means that the nursery was garbage collected by using a special copy algorithm rather than mark and sweep. `tiltratio` indicates what portion of the nursery is used for the allocate subspace. 50% means that the allocate and survivor subspaces are equal in size. However, this ratio is dynamically adjusted upward if mostly short-lived objects are being allocated. `tenureage` indicates the number of “flips” an object must survive before it is promoted to the tenured space.

Within the `<gc-end.../gc-end>` section, you can see the size of the nursery and tenure parts of the heap after the scavenge operation. In this example, the nursery now has **5412152 free bytes**, or **82%** of the nursery space is available.

When the garbage collection cycle caused by the allocation failure is complete, an **allocation-satisfied** tag is generated. This tag indicates that the allocation that caused the failure is now complete.

__ h. Scroll down through the log and locate the different sections and values that are described in the information box. How many allocation failures occurred during startup?

__ i. When you are done examining the `native_stderr.log` file, close it.

__ 5. Rename and save the native_stderr.log.

__ a. Stop server1 from the administrative console.

__ b. Navigate to: `<profiles_root>/profile1/logs/server1`

__ c. Rename the `native_stderr.log` file to: `genconGC.log`

__ d. Move the `genconGC.log` file to: `/usr/Student/GC_Policies`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 2: Change the GC policy for an application server

In this section, you use the administrative console to change the GC policy for server1.

__ 1. Change the GC policy to **optthruput** for server1.

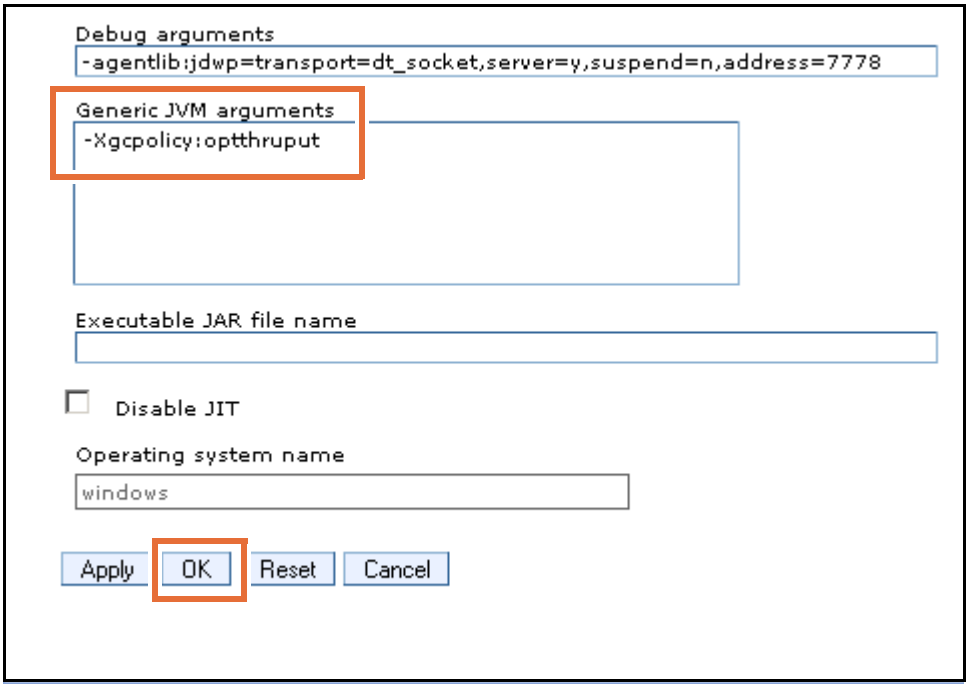
__ a. From the administrative console click **Servers > Server Types > WebSphere application servers > server1**.

__ b. On the configuration tab for server1, under the **Server Infrastructure** section, click **Java and process management**.

__ c. Click **Process definition > Java Virtual Machine**.

__ d. On the JVM configuration tab, scroll down to **Generic JVM arguments**.

__ e. Enter: `-Xgcpolicy:optthruput`



__ f. Click **OK**.

__ g. Click the link to **Save** directly to master configuration.

__ h. Wait for the nodes to synchronize and click **OK**.

__ i. Start **server1** from the administrative console and wait for it to start successfully.

Information

To configure the “optimize for average pause times GC policy”, you would use `-Xgcpolicy:optavgpause` as the generic JVM argument.

To configure the “balanced GC policy”, you would use `-Xgcpolicy:balanced` as the generic JVM argument.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Examine the optthruput verbose GC data for server1.

__ a. Use a Terminal window to navigate to: *<profiles_root>*/profile1/logs/server1

__ b. Open `native_stderr.log` with gedit.

__ c. Examine the text at the top of the log.

```
<initialized id="1" timestamp="2014-02-03T12:27:06.525">
  <attribute name="gcPolicy" value="-Xgcpolicy:optthruput" />
  <attribute name="maxHeapSize" value="0x10000000" />
  <attribute name="initialHeapSize" value="0x3200000" />
  <attribute name="compressedRefs" value="true" />
  <attribute name="compressedRefsDisplacement" value="0x0" />
  <attribute name="compressedRefsShift" value="0x0" />
  <attribute name="pageSize" value="0x1000" />
  <attribute name="pageType" value="not used" />
  <attribute name="requestedPageSize" value="0x1000" />
  <attribute name="requestedPageType" value="not used" />
  <attribute name="gcthreads" value="1" />
  <attribute name="numaNodes" value="0" />
```

__ d. Notice that the GC policy is “optimize for throughput” (optthruput).

__ e. You see a number of allocation failures.

```
<af-start id="3" totalBytesRequested="24" timestamp="2014-02-03T12:27:11.549" intervalms="5023.841" />
<cycle-start id="4" type="global" contextid="0" timestamp="2014-02-03T12:27:11.549" intervalms="5023.870"
/>
<gc-start id="5" type="global" contextid="4" timestamp="2014-02-03T12:27:11.549">
  <mem-info id="6" free="2621440" total="52428800" percent="5">
    <mem type="tenure" free="2621440" total="52428800" percent="5">
      <mem type="soa" free="0" total="49807360" percent="0" />
      <mem type="loa" free="2621440" total="2621440" percent="100" />
    </mem>
  </mem-info>
</gc-start>
<allocation-stats totalBytes="49781568" >
  <allocated-bytes non-tlh="1746064" tlh="48035504" />
  <largest-consumer threadName="main" threadId="000000001174F100" bytes="27737744" />
</allocation-stats>
<gc-op id="7" type="mark" timems="63.357" contextid="4" timestamp="2014-02-03T12:27:11.612">
  <trace-info objectcount="189044" scancount="163501" scanbytes="5250696" />
  <finalization candidates="732" enqueued="342" />
  <ownableSynchronizers candidates="192" cleared="0" />
  <references type="soft" candidates="332" cleared="0" enqueued="0" dynamicThreshold="32"
maxThreshold="32" />
  <references type="weak" candidates="14855" cleared="92" enqueued="46" />
  <stringconstants candidates="4894" cleared="8"  />
</gc-op>
<gc-op id="8" type="sweep" timems="3.410" contextid="4" timestamp="2014-02-03T12:27:11.619" />
<gc-end id="9" type="global" contextid="4" durationms="70.349" timestamp="2014-02-03T12:27:11.619">
  <mem-info id="10" free="35476816" total="52428800" percent="67">
    <mem type="tenure" free="35476816" total="52428800" percent="67">
      <mem type="soa" free="32855376" total="49807360" percent="65" />
      <mem type="loa" free="2621440" total="2621440" percent="100" />
    </mem>
    <pending-finalizers system="342" default="0" reference="46" classloader="0" />
  </mem-info>
</gc-end>
<cycle-end id="11" type="global" contextid="4" timestamp="2014-02-03T12:27:11.620" />
<allocation-satisfied id="12" threadId="0000000011A08F00" bytesRequested="24" />
```

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

<af-end id="13" timestamp="2014-02-03T12:27:11.620" />

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

Interpreting the verbose GC data

The `intervalms` provides the amount of time in milliseconds (ms) between allocation failures.

`totalBytesRequested` is the size of the object in bytes that caused the allocation failure.

Within the `<gc-start.../gc-start>` section, you can see the size of the tenured space at the time of the allocation failure. `soa` stands for small object area, a part of the heap on which small objects are allocated. `loa` is large object area. In this example, at the time of the allocation failure, 0% of the soa was available.

The `<gc-op>` output contains subsections that describe operations that are specific to different garbage collection policies, `optthruput` in this example. These subsections might change from release to release, when improvements are made to the technology or when new data becomes available. `Global` means that the entire heap was garbage collected. In this example, there is a `mark` phase, a `classunload` phase, and a `sweep` phase.

Within the `<gc-end.../gc-end>` section, you can see the size of the soa after the garbage collection process. In this example, the `soa` now has `32855376 bytes free` or **65%** of the `soa` is available.

When the garbage collection cycle that is caused by the allocation failure is complete, an `allocation-satisfied` tag is generated. This tag indicates that the allocation that caused the failure is now complete.

__ f. Scroll down through the log and look for compactions that follow the mark and sweep phase, if any occurred. Recall that a compaction occurs after the mark and sweep phase only when not enough free memory is available to allocate the object. How many allocation failures occurred during startup?

__ g. When you are done examining the `native_stderr.log` file, close it.

__ 3. Rename and save the `native_stderr.log` file.

__ a. Stop server1 from the administrative console.

__ b. Navigate to: `<profiles_root>/profile1/logs/server1`

__ c. Rename the `native_stderr.log` file: `optthruputGC.log`

__ d. Move the `optthruputGC.log` file to: `/usr/Student/GC_Policies`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 3: Use IBM Monitoring and Diagnostic Tools for Java – Garbage Collection and Memory Visualizer (GCMV) to analyze verbose GC data

In this section, you use the GCMV tool to analyze the GC data from each GC policy and compare the behavior between optthruput and gencon. The GCMV tool parses and plots J9 and Sovereign verbose GC logs and -Xtgc output (and is extensible to parse and plot other forms of input). It provides a graphical display of a wide range of verbose GC data values and handles optthruput, optavgpause, and gencon GC modes.

__ 1. Make sure that IBM Support Assistant is started. You started it in the previous sections, but if you stopped it, follow these steps:

__ a. Start the IBM Support Assistant by clicking **Applications** on the toolbar. Select **Start IBM Support Assistant 5**.

__ b. A terminal window pops up. Wait until you see a message that says the IBM Support Assistant is started. Do not close this window until you are finished with the IBM Support Assistant.

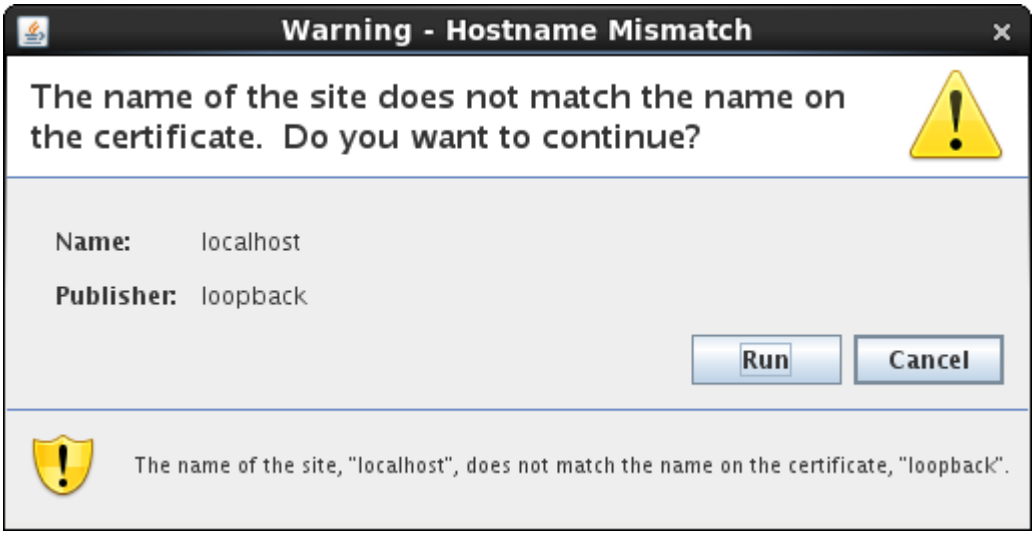
__ 2. Launch the **Garbage Collection and Memory Visualizer (GCMV) [Desktop]** from the **IBM Support Assistant**.

__ a. Open a web browser, and enter the web address:

__ a. `http://washost:10911/isa5`

__ b. You can launch the **GCMV [Desktop]** from the **Tools** tab of the IBM support Assistant.

__ c. Click **Run** if prompted about a hostname mismatch.



__ d. Click **Run**.

__ e. Wait for Java Web Start to download the application and for the GCMV tool to start.

__ 3. Import the GC logs.

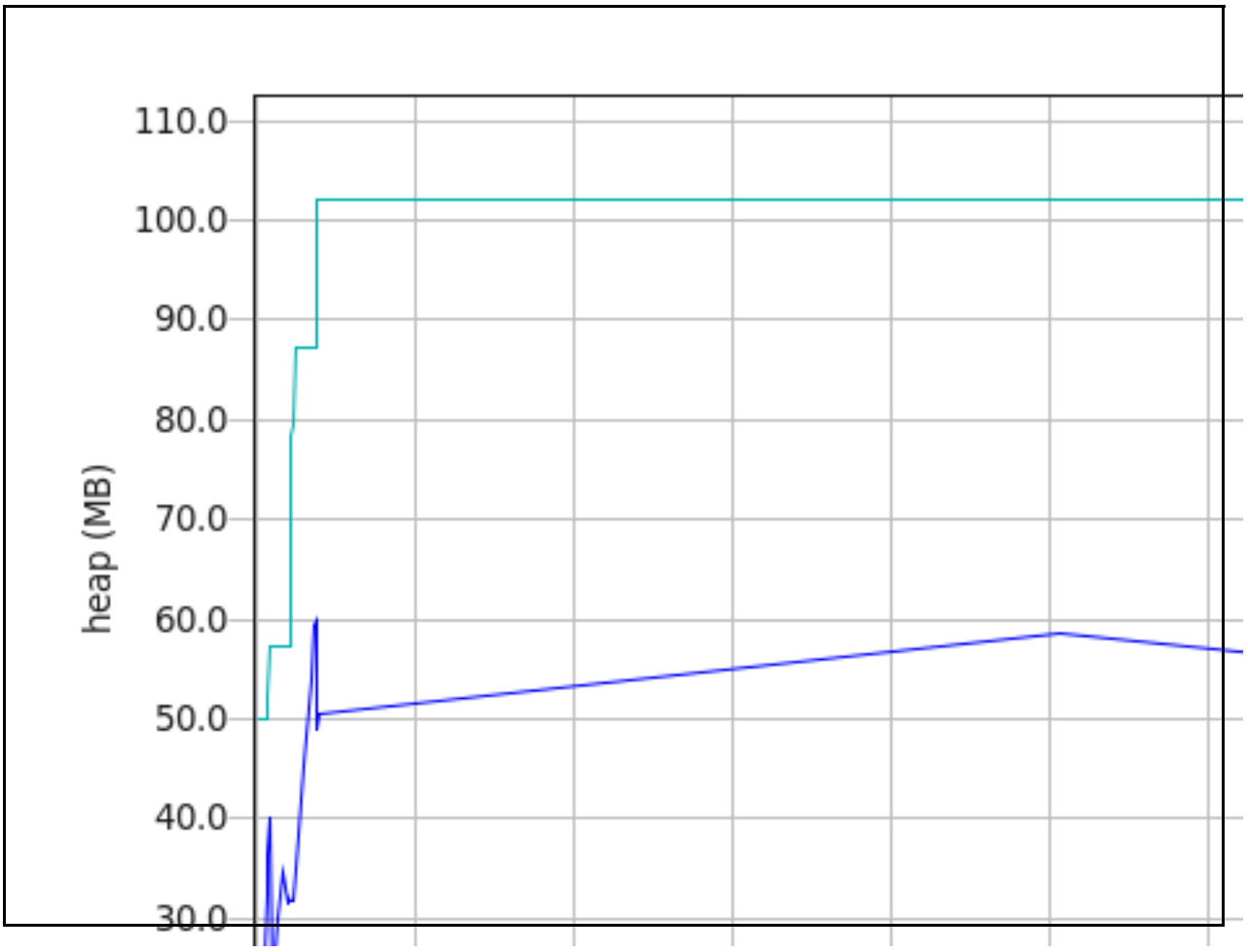
__ a. In the GCMV tool, click **File > Load file**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Navigate to the `/usr/Student/GC_Policies` directory and select `optthruputGC.log`.

__ c. Click **OK**.

__ d. Wait a few moments for the tool to analyze the verbose GC data. You then see a line plot similar to the following graph:



__ e. Recall that this data was logged during the startup of the application server. What you see in the first minute is the initial heap size that starts at 50 MB (the default) and expands suddenly to over 100 MB. The amount of used heap increases from near zero to about 60 MB, which reflects the loading of server and application classes into the heap.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

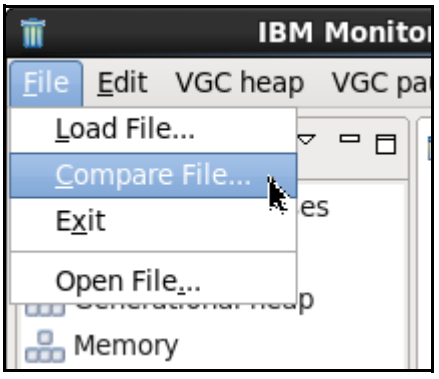
__ f. Select the **Report** tab, and examine the data.

Summary

Forced collection count	3
GC Mode	optthruput
Largest memory request (bytes)	262152
Mean garbage collection pause (ms)	280
Mean heap unusable due to fragmentation (MB)	0.77
Mean interval between collections (ms)	28626
Number of collections	29
Number of collections triggered by allocation failure	26
Proportion of time spent in garbage collection pauses (%)	1.01
Proportion of time spent unpaused (%)	98.99
Rate of garbage collection (MB/minutes)	63.2

__ 4. Analyze the gencon GC data and compare with the optthruput data.

__ a. In the GCMV tool, click **File > Compare File**.



__ b. Browse to `/usr/Student/GC_Policies` and select `genconGC.log`.

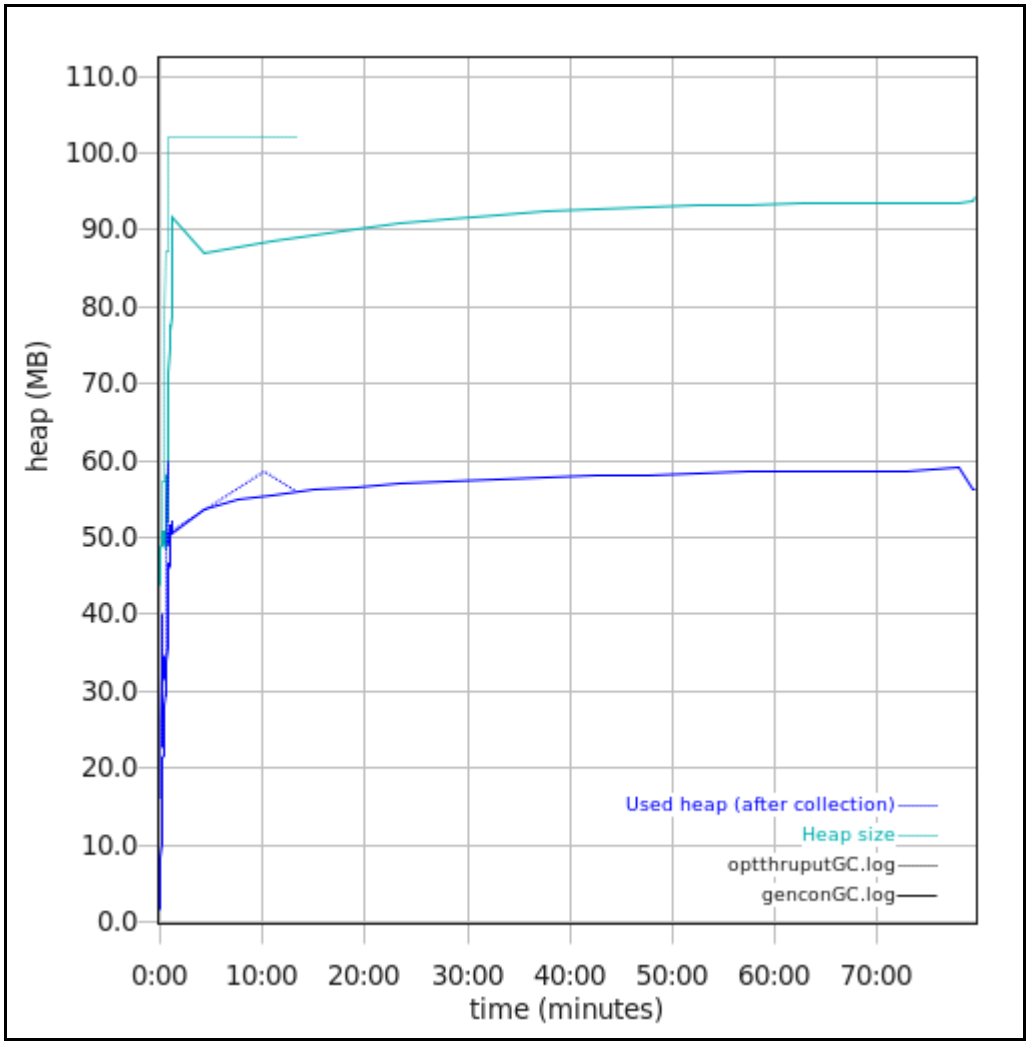
__ c. Click **OK**.

__ d. Wait until the gencon GC data is analyzed.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 5. Compare the two GC policies.

__ a. Select the **Line plot** tab. You see data for each policy plotted in a graph.



The pair of lines in the top of the graph show the heap size, and the pair of lines in the bottom of the graph show the used heap.

The optthruput data is the solid line. The gencon data is the dotted line.

__ b. Select the **Report** tab to view a comparative analysis of the data for each GC policy.

__ c. Examine the complete report that includes:

- Tuning recommendation
- Summary
- Heap size
- Used heap

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Notice in particular the tuning recommendation for each GC policy.

Tuning recommendation

Variant	Tuning recommendation
genconGC.log	1 global garbage collects took on average 765% longer than the average nursery collect. If you believe this is abnormally high and unacceptable, consider using the Balanced GC policy for applications deployed on a 64-bit platform with a heap size greater than 4GB. A high proportion of the nursery is tenured during each collection, possibly leading to longer nursery collections and more frequent collections in the tenured area. Consider increasing the nursery size, to reduce this proportion (the average is 2%).
optthruputGC.log	At one point 944 objects were queued for finalization. Using finalizers is not recommended as it can slow garbage collection and cause wasted space in the heap. Consider reviewing your application for occurrences of the finalize() method. You can use the ISA Tool Add-on, IBM Monitoring and Diagnostic Tools for Java - Memory Analyzer to list objects that are only retained through finalizers. Your application is allocating many large objects, which affects performance.Consider increasing -Xloaminimum and -Xloainitial. The garbage collector is performing system (forced) GCs. 3 out of 29 collections (10.345%) were triggered by System.gc() calls. The use of System.gc() is generally not recommended since they can cause long pauses and do not allow the garbage collection algorithms to optimise themselves. Consider inspecting your code for occurrences of System.gc().

__ e. Notice that the analysis of the optthruput data suggests configuring the balanced GC policy.

Note

Your data for both GC policies might be slightly different from what is shown in the screen captures. In any case, this exercise is meant to show you how the Garbage Collection and Memory Visualizer (GCMV) tool can be used to analyze and compare different GC policies.

__ f. Close the GCMV tool by clicking **File > Exit**.

Section 4: Configure and use the Java Health Center

In this section, you use the Java Health Center to attach to a running JVM and monitor the performance of two GC policies. The IBM Monitoring and Diagnostic Tools for Java – Health Center is a lightweight tool that monitors active IBM virtual machines for Java with minimal effect on

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

performance. The Health Center suggests tuning recommendations for garbage collection, profiles methods and call stacks, and highlights contended locks.

__ 1. Add the generic JVM argument for the Java Health Center.

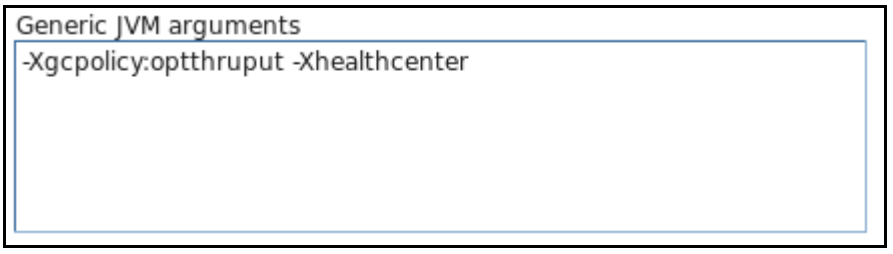
__ a. From the administrative console click **Servers > Server Types > WebSphere application servers > server1**.

__ b. On the configuration tab for server1, click **Java and process management**.

__ c. Click **Process definition > Java Virtual Machine**.

__ d. On the JVM configuration tab, scroll down to **Generic JVM arguments**.

__ e. Enter the argument `-Xhealthcenter` in the box for **Generic JVM arguments** after the `-Xgcpolicy:optthruput` argument as shown; do not type a carriage return.



__ f. Click **OK**.

__ g. Click the link to **Save** directly to master configuration.

__ h. Wait for the nodes to synchronize and click **OK**.

__ i. Restart **server1** from the administrative console and wait for it to start successfully.

Troubleshooting

If server1 fails to start, it is likely that you mistyped the JVM argument. Retype the argument and try starting server1 again.

__ 2. Launch the Java Health Center from the IBM Support Assistant.

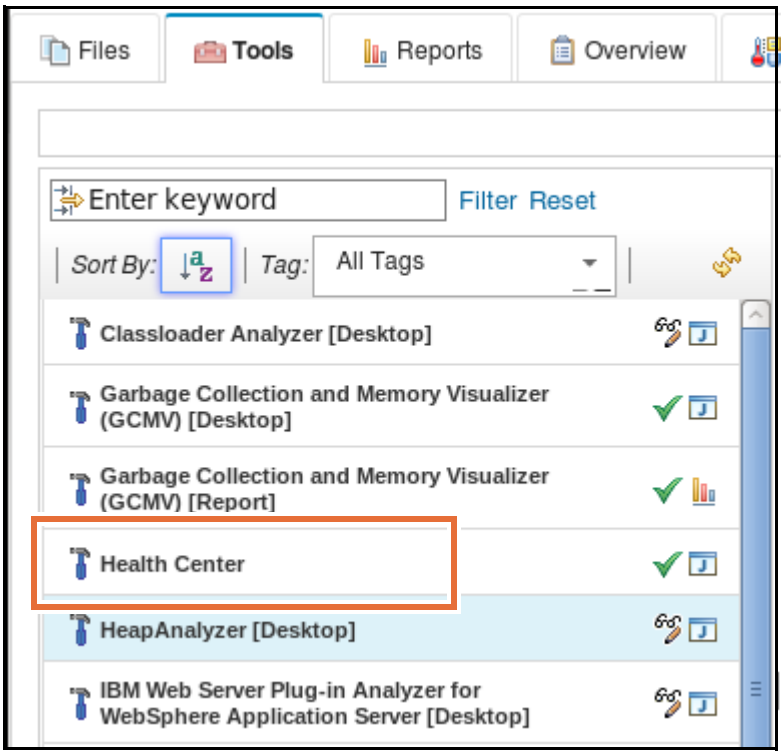
__ a. Open a web browser, and enter the web address:

`http://washost:10911/isa5`

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Wait for the IBM Support Assistant home page to show up in the browser. Click the **Tools** tab.



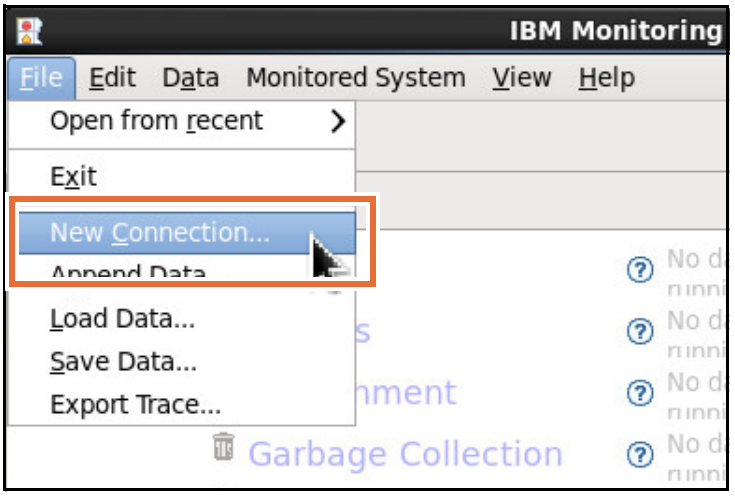
__ c. Select **Health Center** from the list of tools.

__ d. In the right pane for Health Center, click **Launch**.

__ e. On the Run Tool window, click **Launch**.

__ f. Select **Open with javaws**, and click **OK**. Wait until the Health Center opens.

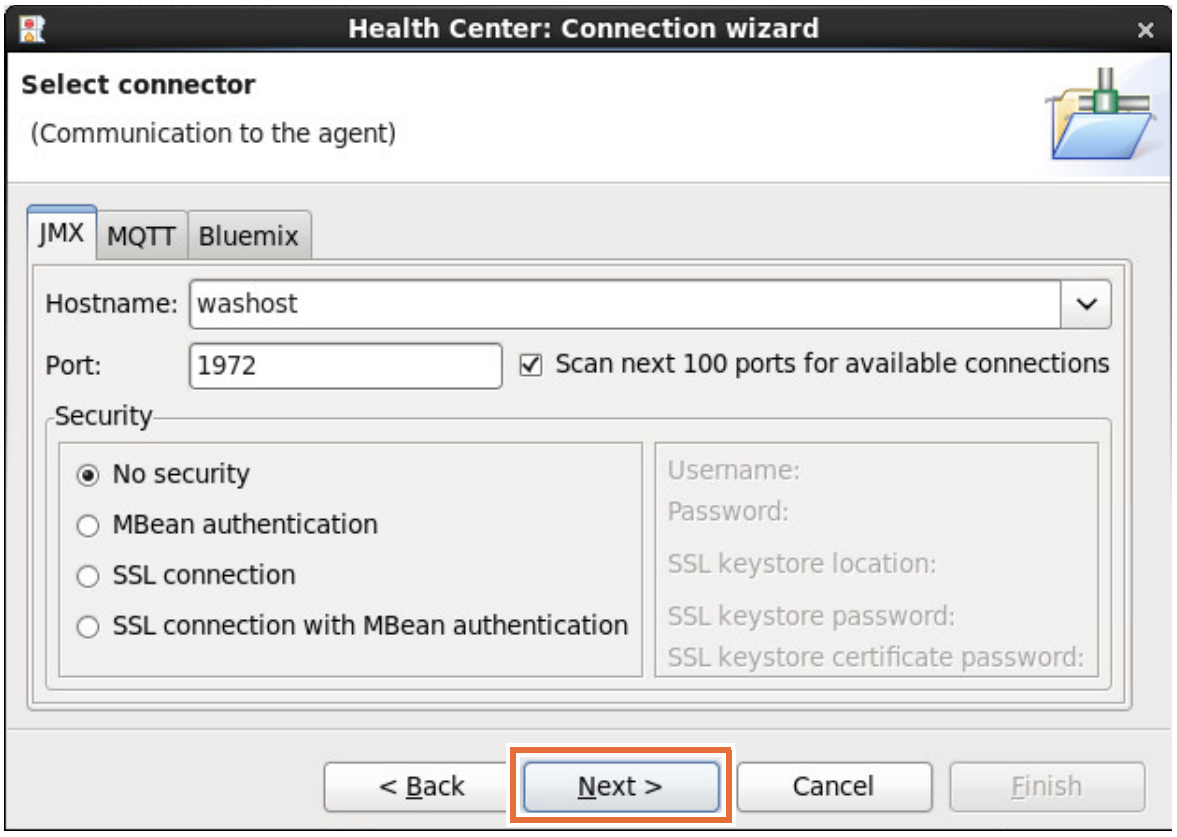
__ g. When the Health Center opens, click **File > New Connection**.



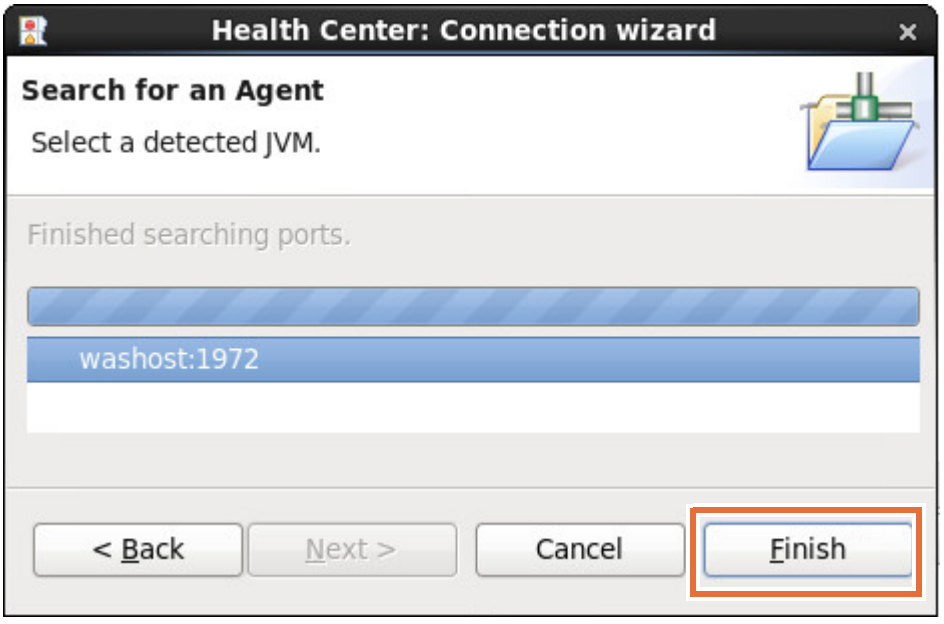
__ h. When the Connection wizard opens, click **Next**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ i. Make the JMX tab is selected, accept the defaults for Hostname and Port, and click **Next.**



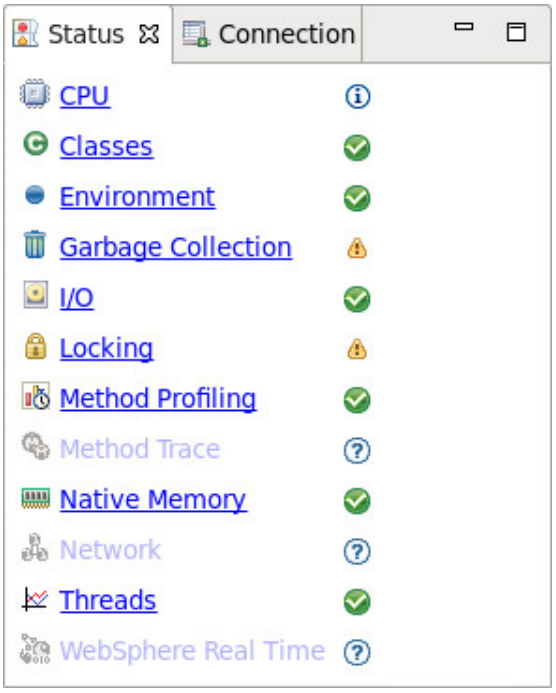
__ j. Click **Finish** as soon as it detects the agent of server1 on port 1972. You do not have to wait for the search to complete.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ k. Wait a few minutes until the Status screen is populated. Notice that the status icons are dynamically updated with data gathered from the running server. Your status might be slightly different.



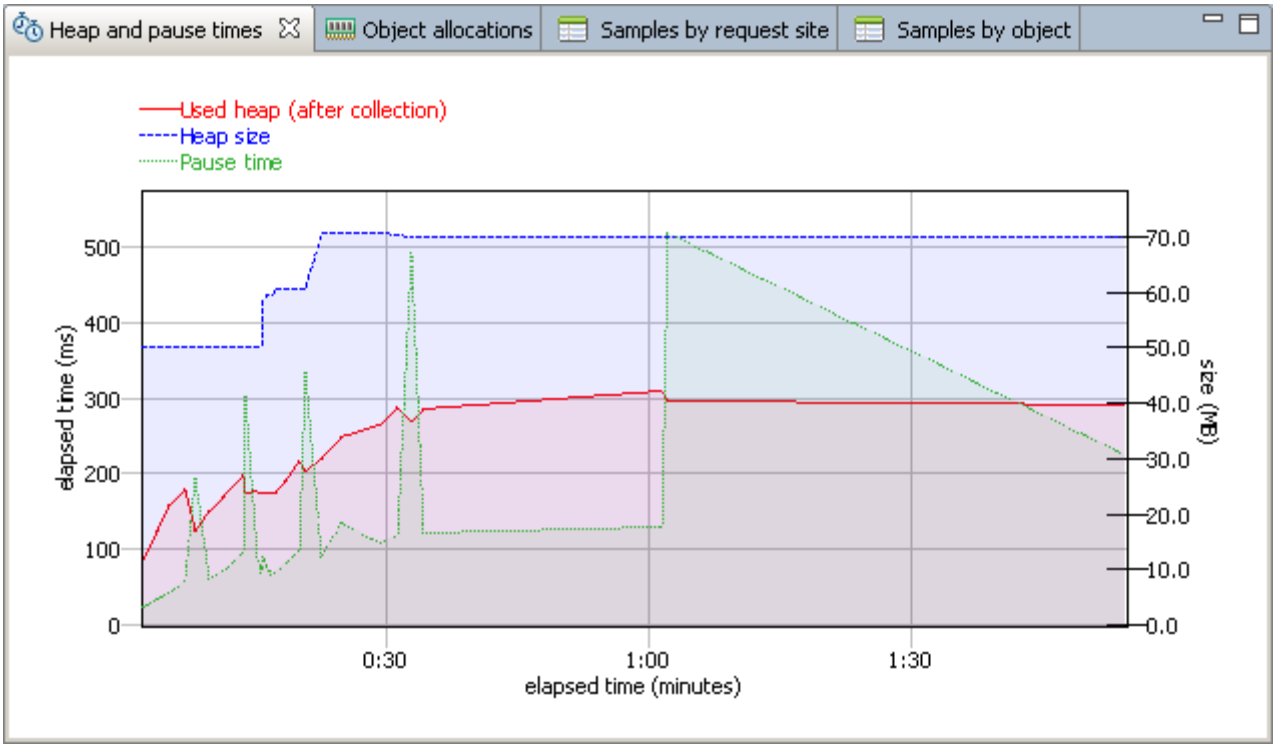
__ l. In addition to garbage collection, the Health Center also monitors class loading, environment, and locking of server resources. Java method profiling is also available. Try clicking any of the links: **Classes**, **Environment**, and **Threads**. Explore what runtime data is provided.

__ 3. Monitor garbage collection.

__ a. Click the **Garbage Collection** link.

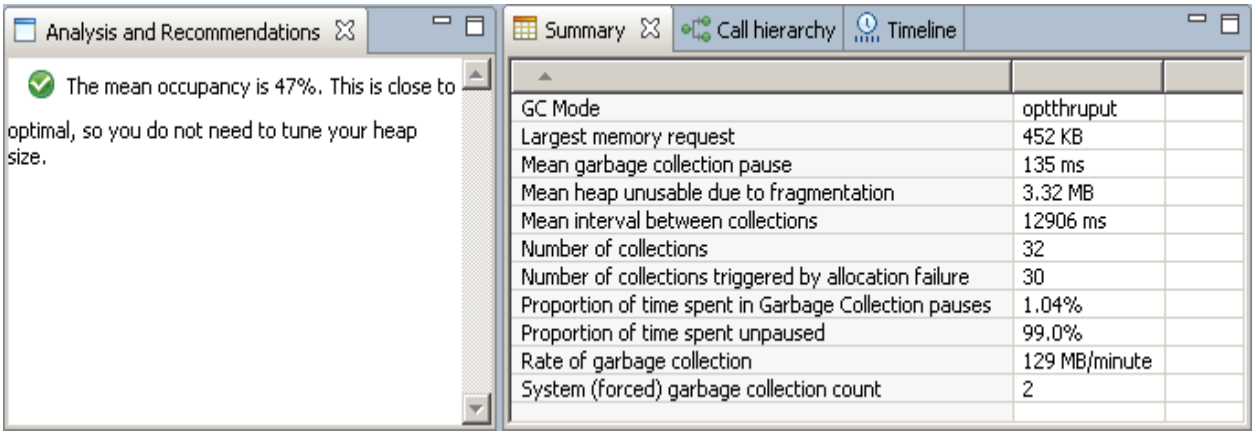
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. The JHC parses the verbose GC data and creates two graphs: Used heap and Pause times. In this view, the two graphs are superimposed on each other for comparison.



__ c. Since the JHC is monitoring a running JVM (that is, server1), these graphs are dynamically updated.

__ d. Tuning recommendations are generated and dynamically updated. A summary of GC events and statistics is displayed which is also dynamically updated.



__ e. Your Recommendations and Summary data might be slightly different.

__ f. When you are done exploring the Java Health Center, close it by clicking **File > Exit**.

__ 4. Change the GC policy of server1 back to the default of gencon.

__ a. From the administrative console click **Servers > Server Types > WebSphere application servers > server1**.

© Copyright IBM Corp. 2013

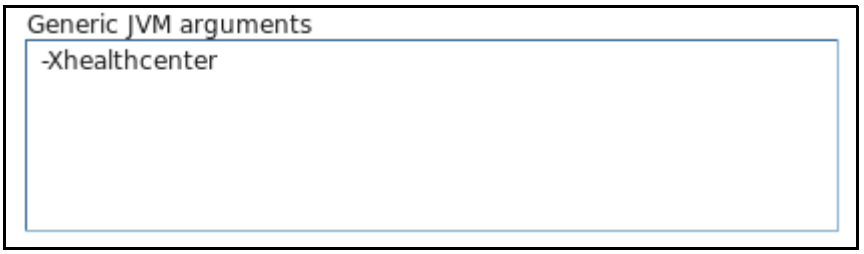
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the configuration tab for server1, under the **Server Infrastructure** section, click **Java and process management**.

__ c. Click **Process definition > Java Virtual Machine**.

__ d. On the JVM configuration tab, scroll down to **Generic JVM arguments**.

__ e. Remove the `-Xgcpolicy:optthruput` argument. You can replace it with `-Xgcpolicy:gencon`, but since this policy is the default, the argument is not necessary.



__ f. Click **OK**.

__ g. Click the link to **Save** directly to master configuration.

__ h. Wait for the nodes to synchronize and click **OK**.

__ i. Restart server1 from the administrative console and wait for it to start successfully.

__ 5. **Optional**: Use the Java Health Center to monitor garbage collection on server1 when the gencon GC policy is configured. You can follow the same steps as you did to monitor the optthruput policy. Look for differences and similarities between the two policies. Pay particular attention to the tuning recommendations.

__ 6. When you are done working with the Java Health Center, close it by clicking **File > Exit**.

__ 7. Keep the IBM Support Assistant running. You are going to use it in the next exercise.

Information

In this exercise, you compared two different GC policies by using the verbose GC data that is generated during startup time. In general, a fast server startup might be a requirement in your environment.

To select the best GC policy for your application server, you must load test your server and analyze the GC data that is generated during the load. Then, based on the tuning recommendations that the GCMV or JHC provides, tune your JVM.

Tuning of the JVM is beyond the scope of this lab exercise. A useful topic, *Tuning the IBM virtual machine for Java*, is in the WebSphere Application Server V8.5 Information Center:

`http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp?topic=%2Fcom.ibm.websphere.nd.doc%2Fae%2Ftprf_tunejvm_v61.html`

End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

This exercise demonstrated how to configure different GC policies for an application server. You configured and examined two different GC policies, optthruput and gencon. You learned how to enable verbose GC for an application server. You examined manually the GC logs that were generated for two different GC policies. In addition, you used the GCMV tool to analyze the GC data and compare the two different GC policies. Finally, you used the Java Health Center tool to monitor garbage collection on a running JVM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 4. Troubleshooting hung threads

What this exercise is about

In this exercise, you detect and troubleshoot hung thread conditions in an application server.

What you should be able to do

At the end of this exercise, you should be able to:

- Use the thread monitor hang detection facility
- Generate and analyze various hung thread scenarios
- Use wsadmin to trigger a javacore file
- Analyze the javacore file for hung threads by using the IBM Thread and Monitor Dump Analyzer for Java tool

Introduction

A common error in Java EE applications is a hung thread. A hung thread can result from a simple software defect (such as an endless loop) or a more complex cause (such as a resource deadlock). This hung transaction consumes system resources, such as processor time, when threads run unbounded code paths, such as when the code is running in an endless loop. Alternatively, a system can become unresponsive even though all resources are idle, as in a deadlock scenario.

Requirements

This exercise requires a special application that is called **BadApp**, which is used to demonstrate different thread problems. You install the application as part of the lab. You use a tool that is called IBM Thread and Monitor Dump Analyzer. This tool is preinstalled and is accessed from the IBM Support Assistant.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Increase the maximum heap size for the deployment manager

Beginning in Exercise 4 and continuing through Exercises 5 and 6, extra load is placed on the deployment manager. To avoid out-of-memory conditions and degraded performance, the maximum heap size for the deployment manager must be increased from its default value of 256 MB to 640 MB.

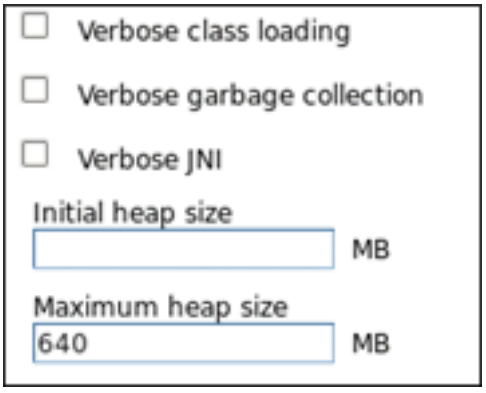
__ 1. Log in to the administrative console.

__ 2. Click **System administration > Deployment manager**.

__ 3. Select the configuration tab. Under Server Infrastructure, click **Java and Process Management > Process definition**.

__ 4. Under Additional properties, click **Java Virtual Machine**.

__ 5. On the configuration tab, set the maximum heap to **640** MB.



__ 6. Click **OK**.

__ 7. Click **Save** to save the configuration.

__ 8. Wait for the node to synchronize, and click **OK**.

__ 9. Log out of the administrative console.

__ 10. Stop and start the Deployment manager from the command line.

- __ a. From a terminal window, change directory to: `<profiles_root>/Dmgr/bin`
- __ b. Enter the command:

  ```
  ./stopManager.sh -user wasadmin -password web1sphere
  ```

- __ c. Wait for the Deployment manager to stop.
- __ d. Enter the command: `./startManager.sh`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 2: Install the test application

You use a specially written application to examine various scenarios where threads become hung.

__ 1. Ensure that the node agent and server1 are started.

__ a. From a terminal window, change directory to

`<profiles_root>/profile1/bin`

__ b. Enter the command: `./serverStatus.sh -all`

__ c. If the node agent is stopped, start it with command: `./startNode.sh`

__ d. If server1 is stopped, start it with command: `./startServer.sh server1`

__ 2. Install the Badapp application. This application is specially developed to exhibit common behaviors such as hung threads and deadlocks.

__ a. In a terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the following command, all on one line:

```
./wsadmin.sh -f /usr/Student/Hang/install_badapp.py -user wasadmin
-password web1sphere
```

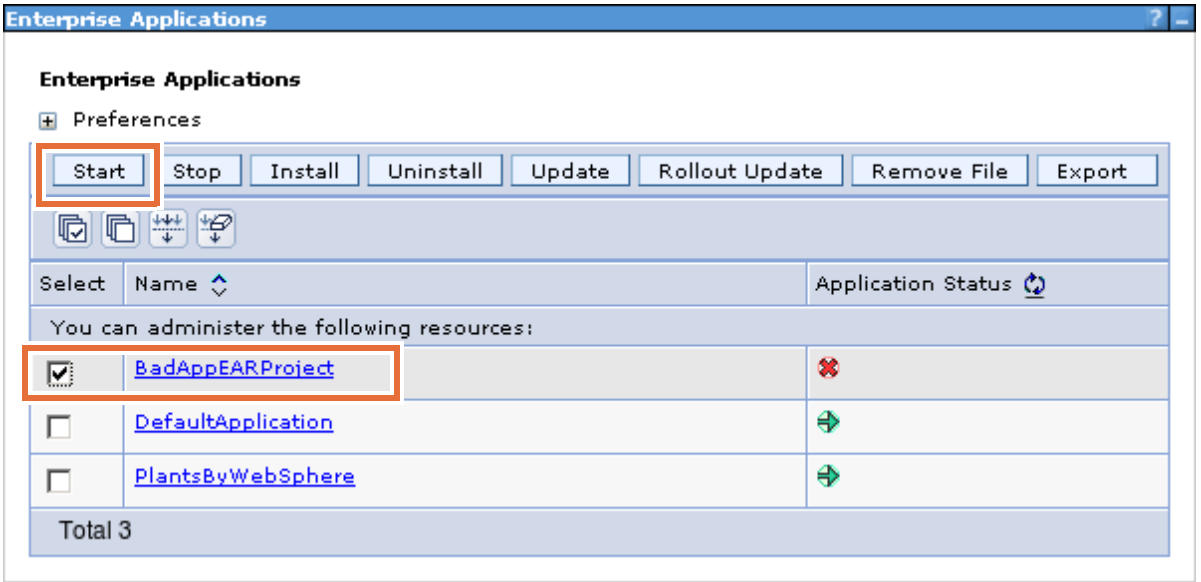
__ c. Wait until you see the following message in the command window.

```
ADMA5013I: Application BadAppEARProject installed successfully.
-----------------------------------------------------------------
 AdminNodeManagement:        Synchronize the active nodes
 Usage: AdminNodeManagement.syncActiveNodes()
 Return: If the command is successfully invoked, a value of 1 is
returned.
-----------------------------------------------------------------
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 3. Verify that the application is installed and start it from the administrative console.

__ a. Log on to the administrative console and click **Applications > Application Types > WebSphere enterprise applications**.



__ b. Select **BadAppEARProject**, and click **Start**.

Section 3: Manually initiate a javacore dump

In this section, you complete the steps to create a thread dump (javacore dump). The javacore file is a key artifact that is used to troubleshoot server hangs.

__ 1. Create a javacore dump by using wsadmin.

__ a. From a command prompt, change the directory to: `<profiles_root>/profile1/bin`

__ b. Start the wsadmin shell by entering:

```
./wsadmin.sh -lang jython -user wasadmin -password web1sphere
```


© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. From the wsadmin prompt, type the following two commands: (if you prefer, copy, and paste these commands from the file `/usr/Student/Hang/wsadmin_cmds.txt`):

```
jvm=AdminControl.completeObjectName('type=JVM,process=server1,*')
AdminControl.invoke(jvm, 'dumpThreads')
```

```
jvm=AdminControl.completeObjectName('type=JVM,process=server1,*')
AdminControl.invoke(jvm, 'dumpThreads')
```

root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin

File Edit View Search Terminal Help

```
[root@washost bin]# ./wsadmin.sh -lang jython -user wasadmin -password web1sphere
WASX7209I: Connected to process "dmgr" on node washostCellManager using SOAP connect
or;  The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>
wsadmin>jvm=AdminControl.completeObjectName('type=JVM,process=server1,*')
wsadmin>
wsadmin>AdminControl.invoke(jvm, 'dumpThreads')
''
wsadmin>
```

__ d. After the command completes, type `quit` to exit the wsadmin shell and close the command prompt window.

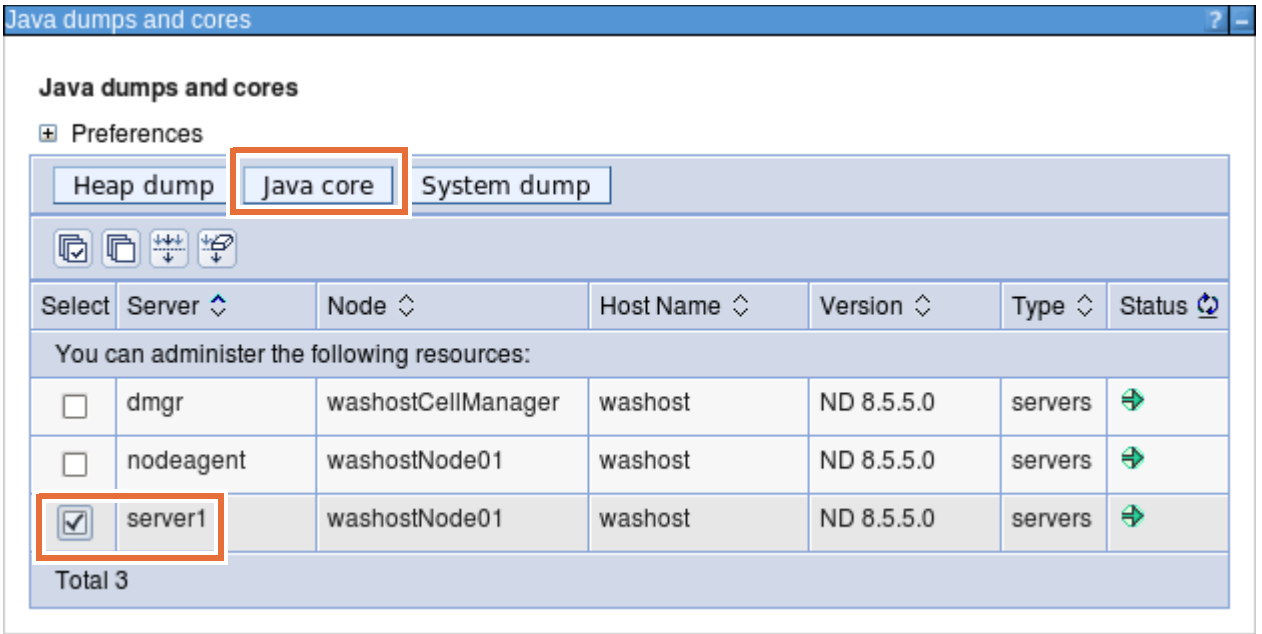
__ e. Look for the javacore dump in `<profiles_root>/profile1`.

__ f. You see a file similar to `javacore.<timestamp>.<PID>.<seq_no>.txt`. You can open it with a text editor if you want, but you are going to examine other javacore files in the next section.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Trigger a javacore dump by using the administrative console.

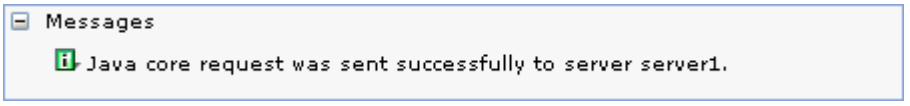
__ a. From the administrative console, select **Troubleshooting > Java dumps and cores**.



__ b. Select **server1**, and click **Java core**.

__ c. You see a console message when the request completes:

```
Java core request was sent successfully to server server1.
```



__ d. Again, look for the javacore dump in *<profiles_root>*/profile1.

Information

The profile directory is the default location for javacore files. This location includes javacore files that are created on demand and those files that are created due to a server crash. If there is more than one present, it is often best to begin the investigation with the file created closest to the time when the problem occurred.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 4: Configure the hang detection policy

Information

Hung thread detection is turned on by default in WebSphere Application Server. In the default configuration, a thread must be active for at least 10 minutes before it is marked as potentially hung. In the interest of time, you lower that threshold to 1 minute by setting a custom property. Next, set another property to change the interval at which the thread monitor polls for hung threads from 3 minutes to 20 seconds. Thirdly, configure the automatic dumping of javacore when the hung thread is detected.

__ 1. In the administrative console, open the Custom Properties panel for your application server.

- __ a. Select **Servers > Server Types > WebSphere application servers > server1**.
- __ b. Under the **Server Infrastructure** area, expand **Administration** and click the link **Custom properties**. You might need to scroll down to see this link.

__ 2. Create a customer property to mark inactive threads as hung threads after 60 seconds.

- __ a. In the **Custom properties** panel, click **New**.
- __ b. In the **Name** field, type: `com.ibm.websphere.threadmonitor.threshold`
- __ c. In the **Value** field, type: `60`
- __ d. Click **OK**.

Information

This property tells the thread monitor to mark any thread active for longer than 1 minute (60 seconds) as hung.

__ 3. Create a second custom property to configure how often WebSphere looks for hung threads.

- __ a. In the **Custom properties** panel, click **New**.
- __ b. In the **Name** field, type: `com.ibm.websphere.threadmonitor.interval`
- __ c. In the **Value** field, type: `20`
- __ d. Click **OK**.

Information

This property tells the thread monitor to look for hung threads every 20 seconds.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. Create a third custom property to tell the thread monitor to generate a javacore when a hung thread is detected.

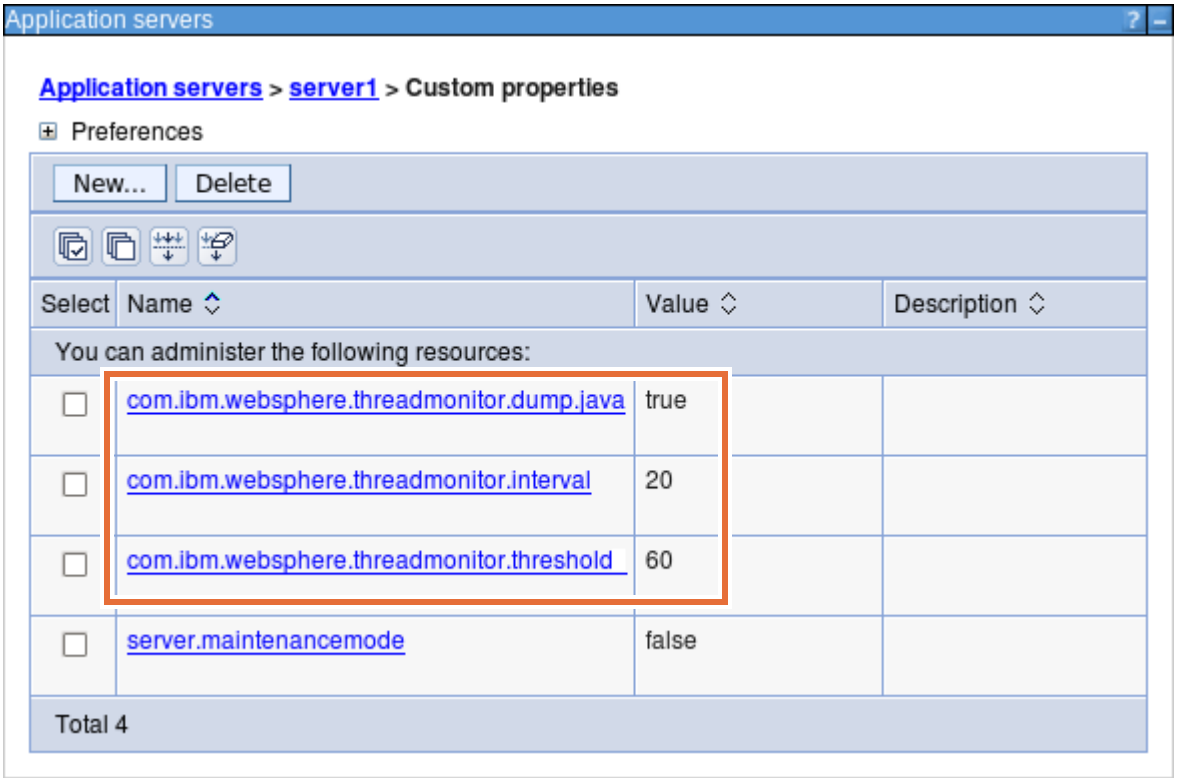
__ a. In the **Custom properties** panel, click **New**.

__ b. In the **Name** field, type: `com.ibm.websphere.threadmonitor.dump.java`

__ c. In the **Value** field, type: `true`

__ d. Click **OK**.

__ 5. After all the properties are created, check the **Custom properties** list:



__ a. Intelligent Management uses the `server.maintenancemode` property, and it is unrelated to the thread monitor. Click **Save** and wait for the nodes to synchronize. Click **OK**.

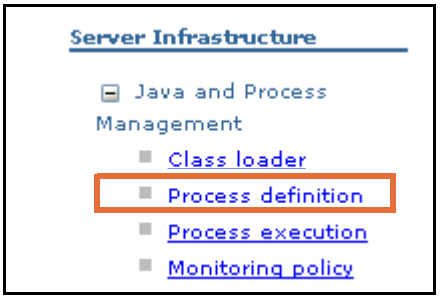
__ 6. Configure an environment property to direct the javacore dumps to a custom location.

__ a. Select **Servers > Server Types > WebSphere application servers > server1**.

 © Copyright IBM Corp. 2013

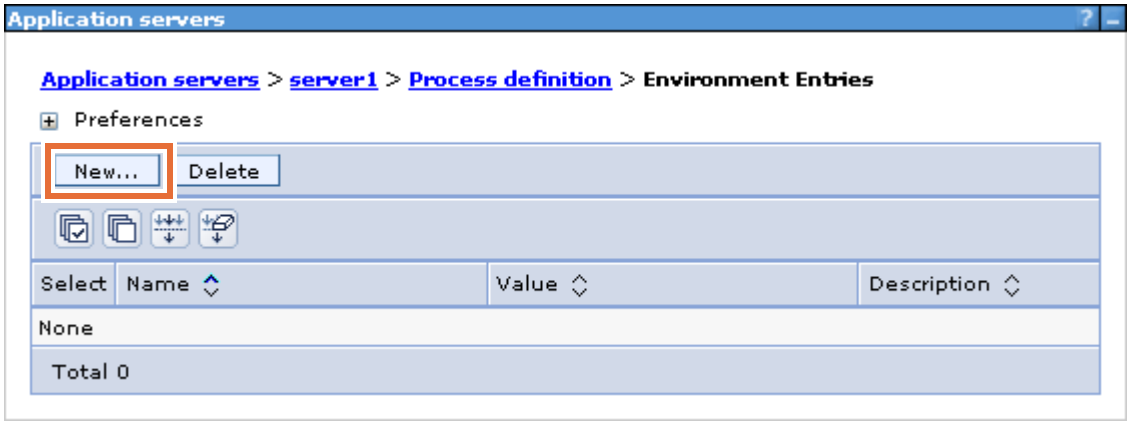
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Under Server Infrastructure, expand **Java Process and Management** and click the **Process definition** link.

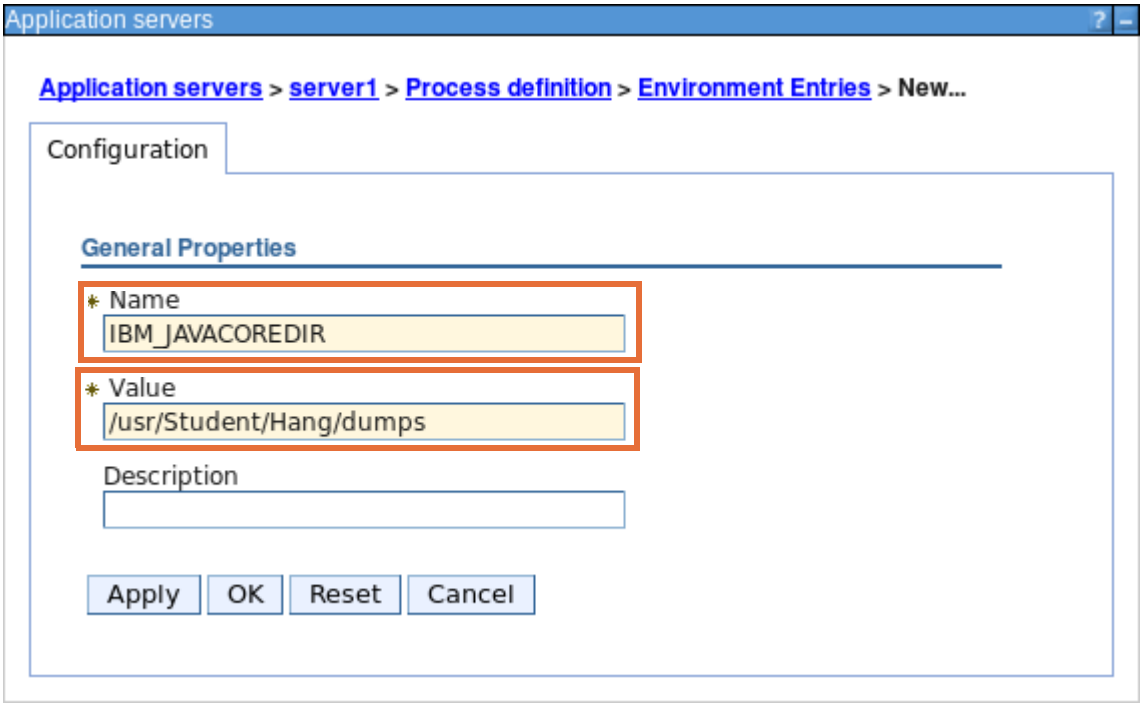


__ c. Under Additional Properties, select **Environment Entries**.

__ d. Click **New**.



__ e. Enter `IBM_JAVACOREDIR` for the name, and `/usr/Student/Hang/dumps` for the value.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Click **OK**.

__ g. Click **Save** and wait for the nodes to synchronize. Click **OK**.

__ h. For these new properties to be effective, the server must be restarted. Restart **server1** from the administrative console.

Information

The environment variable `IBM_JAVACOREDIR` is deprecated as of version 8, though it still works. In your environment, you can replace it with the `-Xdump` JVM generic argument.

For example, the following `-Xdump` argument has the same effect as the `IBM_JAVACOREDIR` environment variable that you defined.

```
-Xdump:java:events=user,range=1..1,file=/usr/Student/Hang/dumps/javacore.%Y%m%d
.%H%M%S.%pid.%seq.txt
```

__ 7. After the server restarts, test that the javacore dump is written to the `/usr/Student/Hang/dumps` directory. You can use the javacore dump button in the administrative console. Notice that if the `dumps` directory did not exist, it is created. After testing, delete the javacore file from this directory.

__ a. From the administrative console, select **Troubleshooting > Java dumps and cores**

__ b. Select **server1**, and click **Java core**.

__ c. Navigate to `/usr/Student/Hang/dumps` and verify that the memory dump is written there.

__ d. Delete the javacore file from the `dumps` directory.

Section 5: Generate and analyze a deadlock scenario

Some application servers' hang conditions result from a deadlock situation where two or more threads are waiting for each other to complete. To detect a deadlock hang, you can generate a thread dump (javacore file) and examine it manually using a text editor or by using the IBM Thread and Monitor Dump Analyzer tool.

Information

Deadlock scenario

In a true deadlock, the web application is able to be called to do at least two different operations; that is to follow two different – fatally different in this case – code paths.

In the first operation, the code path locks some resource (for example, opens a file in exclusive mode or locks a database row for update) and then tries to lock another resource.

In the second operation, the code path locks the second resource and then tries to lock the first resource. The term "locks" versus "tries to lock" is used for a reason.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

The unrecoverable scenario is when the two operations are performed at the same time (presumably by two users, each in their own browser window). User one (running in web container thread 1) successfully locks resource A while user two (running in web container thread 2) successfully locks resource B. These two threads might typically run to completion, first one and then the other.

However, if any code takes a comparatively long time, such as file I/O, thread 1 runs part way through and then thread 2 is allowed to run part way through. In this way, you can end up in the situation described. If user 1 (thread 1) is allowed to run again, it tries to lock resource B. This attempt fails because user 2 (thread 2) already locked resource B. User1 (thread 1) then waits for resource B to be unlocked. User2 (thread 2) is then allowed to run again, and it tries to lock resource A. This attempt also fails because user 1 (thread 1) already locked resource A. User2 (thread 2) then waits for resource A to be unlocked. Both threads are now waiting for the other to complete their code path and unlock a resource, but neither can run. This behavior is the essence of a deadlock.

BadApp produces this error by using two Java objects (of type Object) as the resources, and two methods to do the two patterns of locking. To essentially force the pattern of back-and-forth threads that are running which results in the deadlock error, these methods lock the first resource and then sleep for 30 seconds before trying to lock the second resource.

__ 1. Use the BadApp application to create a deadlock situation.

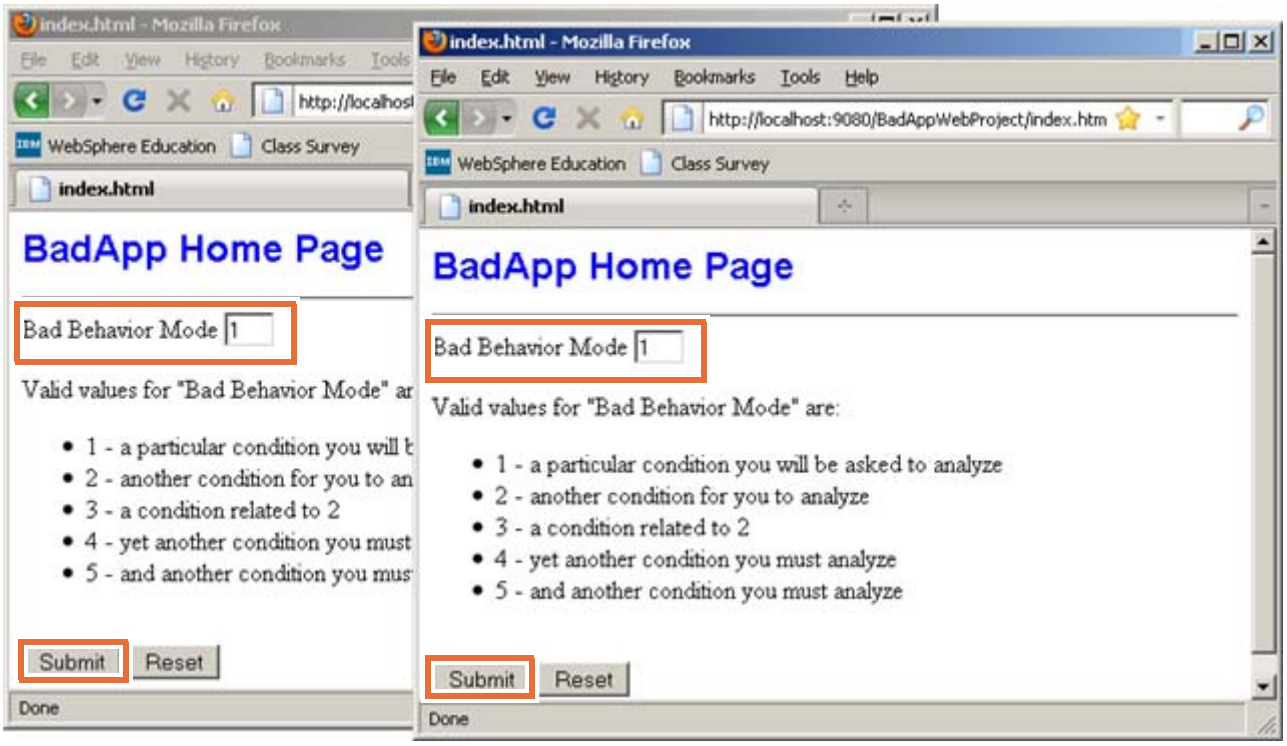
__ a. Open a browser and enter the following web address:

`http://localhost:9080/BadAppWebProject/index.html`

__ b. Bookmark this page, as you are going to use it frequently.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

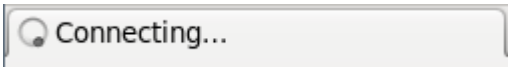
__ c. Open another browser and go to the same address.



__ d. Make sure that the Bad Behavior Mode is **1**. Click **Submit** on the first (top) window.

__ e. Wait **10** seconds; then click **Submit** on the second window.

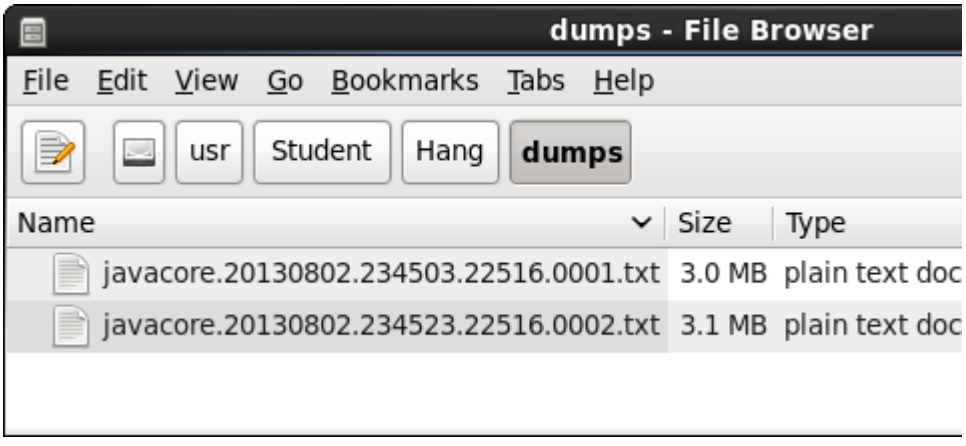
__ f. If successful, neither browser window comes back with any response; the browser "working" icon continues to spin until the request times out with an error.



__ 2. Examine the javacore artifacts of the hang condition.

__ a. Use a file system browser to navigate to the `/usr/Student/Hang/dumps` directory. You see two javacore files. If you do not see two files, wait a few minutes.

__ b. Double-click the first file to open it in gedit. Notice that your sequence numbers might be different from those numbers that are shown in the screen capture



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Search for the string “Deadlock” by using the **Search > Find** feature of gedit.

__ d. You see a message similar to the following text:

```
1LKDEADLOCK Deadlock detected !!!
NULL        ---------------
NULL
2LKDEADLOCKTHR Thread "WebConainer : 1" (0x1675EE00)
3LKDEADLOCKWTR is waiting for:
. . .
```

```
1LKDEADLOCK    Deadlock detected !!!
NULL        ------------------
NULL
2LKDEADLOCKTHR  Thread "WebContainer : 1" (0x1675EE00)
3LKDEADLOCKWTR    is waiting for:
4LKDEADLOCKMON      sys_mon_t:0x1498B410 infl_mon_t: 0x1498B44C:
4LKDEADLOCKOBJ     java/lang/Object@0x05201078
3LKDEADLOCKOWN    which is owned by:
2LKDEADLOCKTHR  Thread "WebContainer : 0" (0x1CF72E00)
3LKDEADLOCKWTR    which is waiting for:
4LKDEADLOCKMON      sys_mon_t:0x14995ED8 infl_mon_t: 0x14995F14:
4LKDEADLOCKOBJ     java/lang/Object@0x05201088
3LKDEADLOCKOWN    which is owned by:
2LKDEADLOCKTHR  Thread "WebContainer : 1" (0x1675EE00)
NULL
NULL        ------------------------------------------------------------------
```

__ e. In this classic deadlock scenario, notice that two web container threads are waiting for each other to release a resource that the other locked. If you search the second javacore file, you see a similar deadlock detected message.

__ 3. Examine the server log messages.

__ a. If HPEL is enabled for server1, use the Log Viewer to examine the logs. From the administrative console, click **Troubleshooting > Logs and trace > server1 > View HPEL logs and trace**. Look for messages from the ThreadMonitor. Try filtering in the **Include loggers** window by using the string `*ThreadMonitor*`.

__ b. If basic logging is enabled for server1, navigate to `<profiles_root>/profile1/logs/server1`, and open the `SystemOut.log` file by using gedit. Look for messages from the ThreadMonitor.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Look for 2 log records similar to the following text:

```
... Thread "WebContainer : 0" ... has been active for 64,516 milliseconds
and may be hung.
. . .
```

```
0000003b ThreadMonitor W   WSVR0605W: Thread "WebContainer : 0"
(0000005d) has been active for 64,516 milliseconds and may be hung.  There is/are
1 thread(s) in total in the server that may be hung.
            at
com.ibm.issf.atjolin.badapp.BadAppServlet.dopeyMethod(BadAppServlet.java:320)
            at
com.ibm.issf.atjolin.badapp.BadAppServlet.doPost(BadAppServlet.java:257)
            at javax.servlet.http.HttpServlet.service(HttpServlet.java:595)
            .
            .
0000003b ThreadMonitor W   WSVR0605W: Thread "WebContainer : 1"
(0000005e) has been active for 60,719 milliseconds and may be hung.  There is/are
2 thread(s) in total in the server that may be hung.
            at
com.ibm.issf.atjolin.badapp.BadAppServlet.sneezyMethod(BadAppServlet.java:337)
            at
com.ibm.issf.atjolin.badapp.BadAppServlet.doPost(BadAppServlet.java:259)
            .
            .
```

__ d. Notice that the two threads are active for greater than 60 seconds (the thread monitor threshold value you set in the previous section). Therefore, the thread monitor issues a warning that the thread is potentially hung. The number of milliseconds that you actually see varies, but is within the range from threadmonitor.threshold to (threadmonitor.threshold+threadmonitor.interval). In this exercise, the range is 60000–80000 milliseconds.

__ e. Also, notice that the stack trace for each suspected hung thread is printed. This information can be used to identify the possible cause for the hang. In this example, it seems that the **dopeyMethod** and the **sneezyMethod** are involved in the deadlock.

__ 4. Resolve the deadlock in the running server.

__ a. Check back on the two browser windows. If they timed out or if they are still spinning, exit the browsers.

__ b. However, the threads remain deadlocked, which you can see when you trigger another thread dump. You must restart the server to release the threads. Restart **server1**.

__ c. The problem is clearly with the application itself, and must be returned to the application developers to resolve.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

Interrupting threads in a running servers.

Java does not provide a general mechanism to forcibly interrupt threads, and WebSphere does not currently provide a way to terminate threads in a running JVM.

A high-end monitoring tool such as IBM Tivoli Composite Application Manager Application Diagnostics can be used to suspend hung threads.

Section 6: Generate and investigate hung application threads

In this section, you explore when an application causes threads to wait for a comparatively long time, the application seems to hang while waiting for locked resources.

__ 1. Remove the javacore files from the last section.

__ a. Navigate to the `/usr/Student/Hang/dumps` directory and delete all the javacore files.

Important

Practice the steps in this section

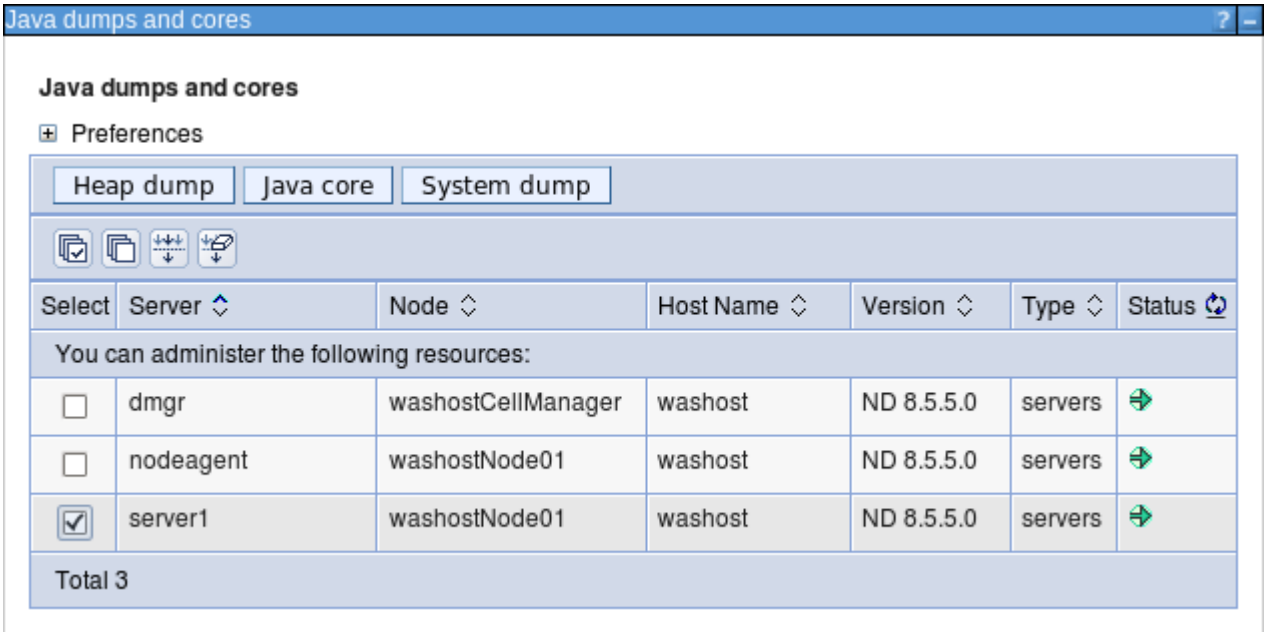
Read this section first before performing the steps. You work with multiple web browsers (web container threads) to create a blocked threads condition. You trigger two javacore files: one while the threads are blocked, and the other after some or most threads complete.

You might consider doing this section twice to make sure that you get the intended results.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Generate another hung threads condition and trigger several javacore dumps.

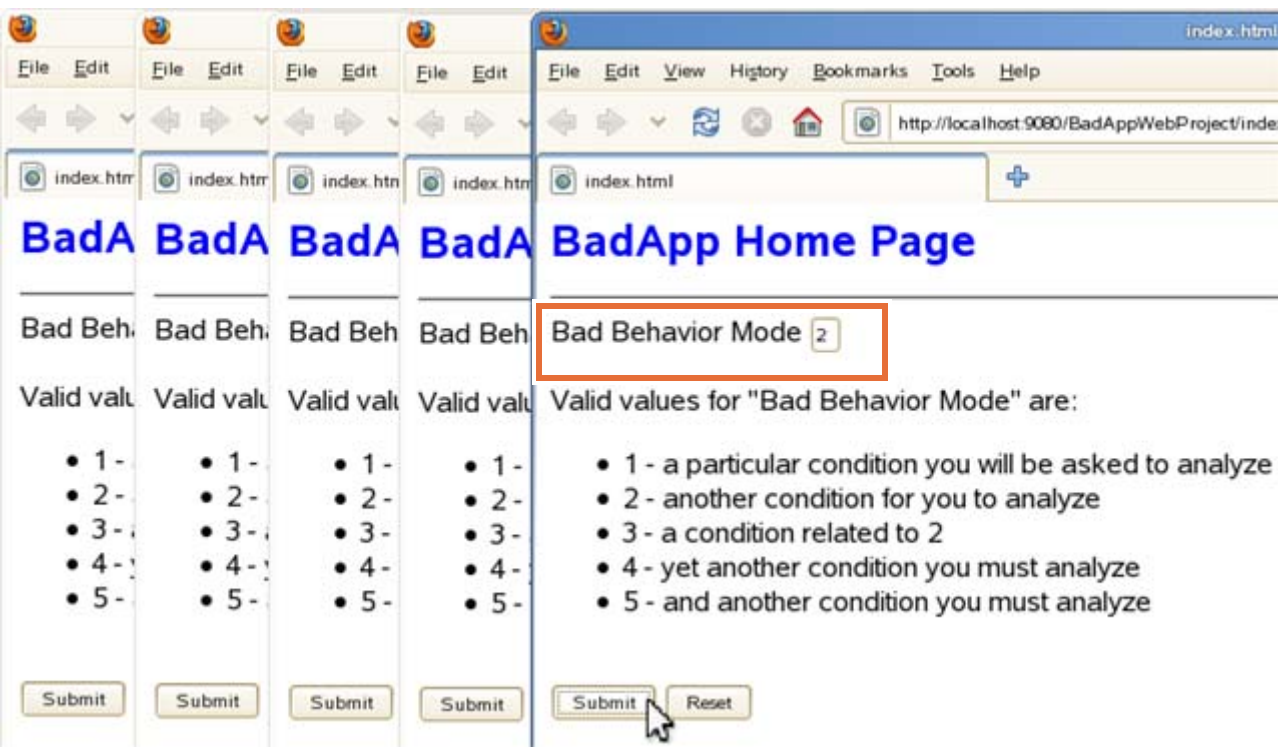
__ a. Prepare to trigger javacore dumps by navigating to the Java dumps and cores pane. Select **Troubleshooting > Java dumps and cores**.



__ b. Select **server1**, but do not click **Java core** now.

__ c. Open **five** browser windows and browse to the **BadApp home page** in each.

__ d. In each browser window, type **2** in the **Bad Behavior Mode** field but ***do not*** click **Submit** yet.



© Copyright IBM Corp. 2013

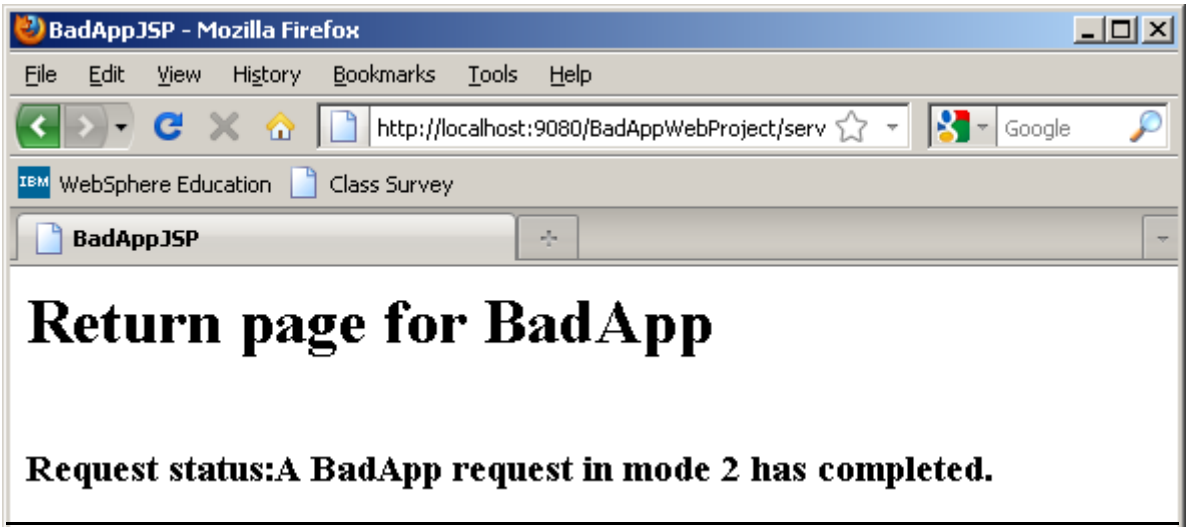
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. When all browser windows have **2** entered, click **Submit** on each window as rapidly as you can. **HINT**: It helps if you position the windows so that they overlap, but the Submit button is visible in each one; ***and*** you click the top-most window first, then the next, and so on.

__ f. As quickly as you can after clicking the last **Submit** button (ideally within 8 seconds, at most within 20 seconds), trigger a javacore dump by clicking **Java core dump** in the administrative console.

__ g. Watch the browser windows. You see a 40-second overall delay before the last browser window returns. You see the following message:

```
Request status: A BadApp request in mode 2 has completed.
```



__ h. As soon as the last browser shows this message, generate another javacore dump by clicking **Java core dump** on the administrative console.

__ i. Close all the BadApp web browser windows.

__ j. Your next step is to use the Thread and Dump Monitor tool from the IBM Support Assistant to compare the two javacore files that you generated in this section.

__ 3. Start the IBM Support Assistant.

__ a. From your system toolbar, click **Applications**.

__ b. Select **IBM Support Assistant > Start IBM Support Assistant 5**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

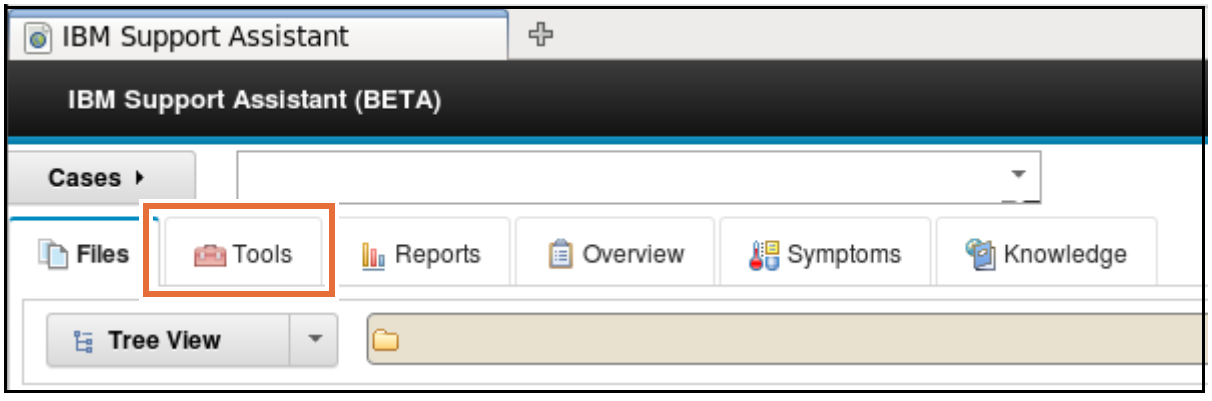
__ c. A terminal window pops up that shows the startup of IBM Support Assistant. Wait for IBM Support Assistant to start.

__ 4. Open the IBM Support Assistant and start the Thread and Monitor Dump Analyzer tool.

__ a. Start a new web browser window, and enter the web address:

`http://washost:10911/isa5`

__ b. Click the **Tools** tab.



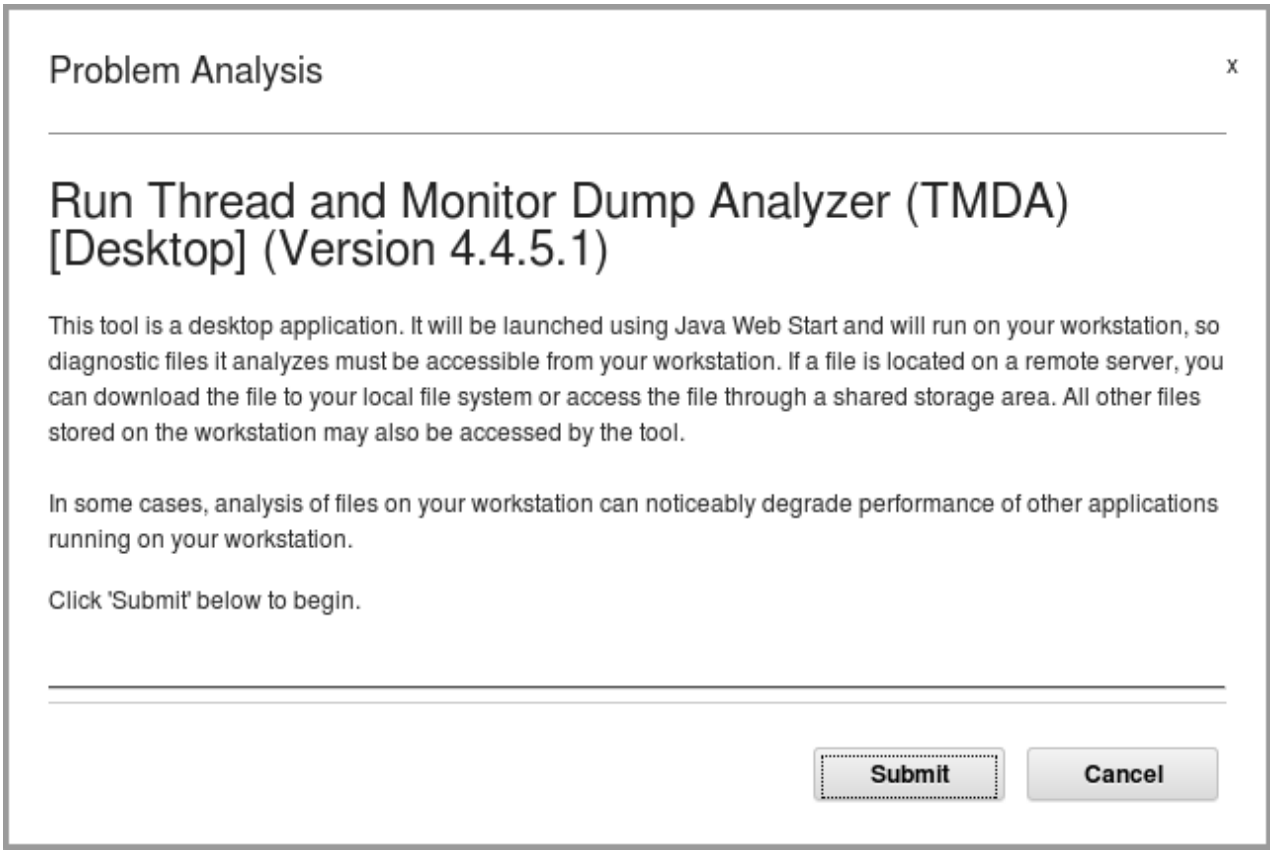
__ c. Scroll down the list of tools and click the **TMDA [Desktop]** tool.

__ d. In the right pane of IBM Support Assistant, click **Launch** for TMDA [Desktop].

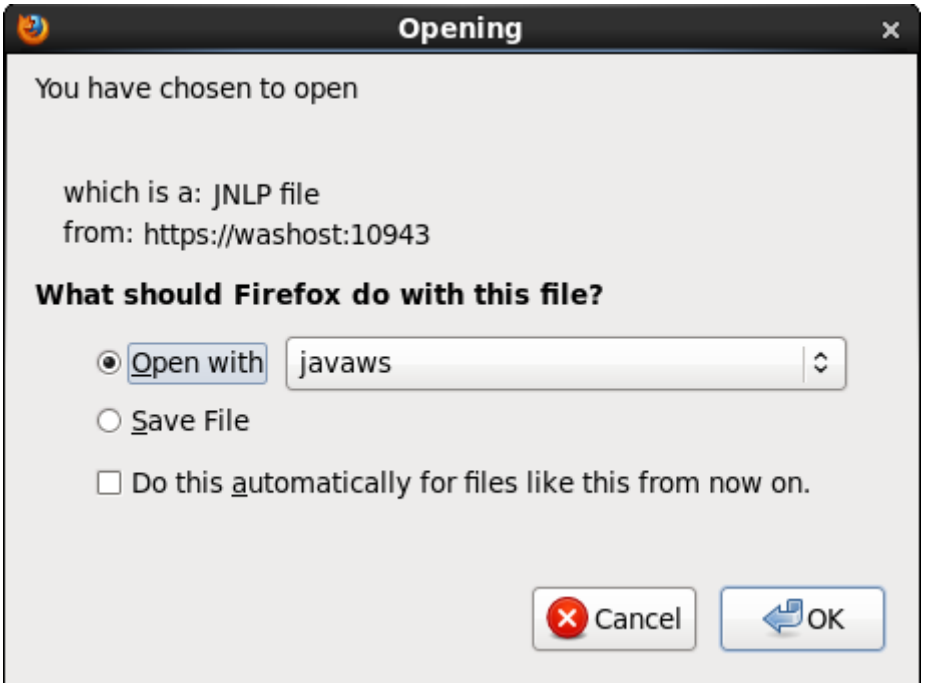
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Another window pops up; click **Submit**.



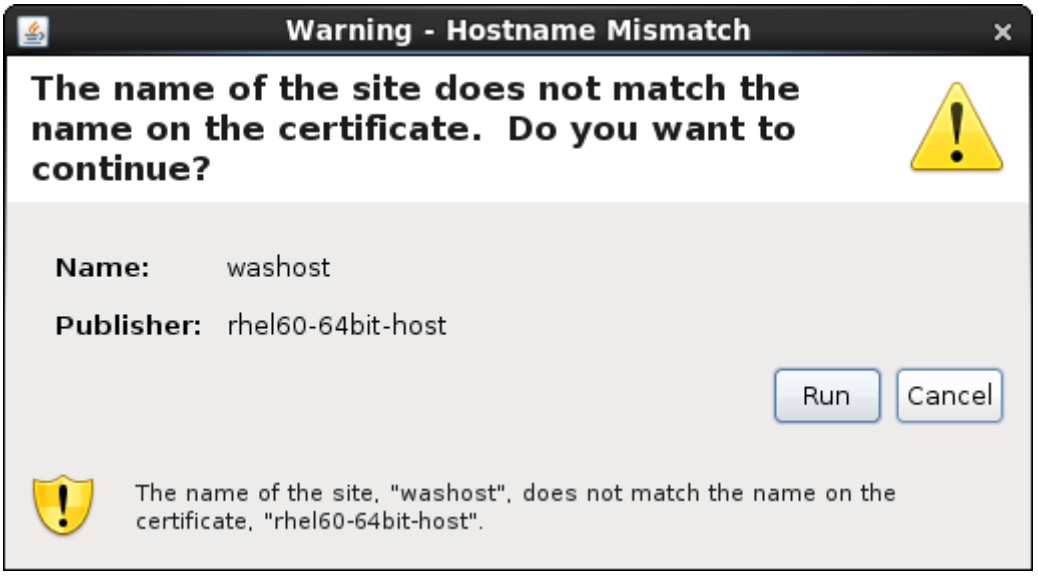
__ f. On the next window, select **Open with javaws**, and click **OK**.



__ g. If javaws is not listed in the menu for the “Open with” option, browse your file system for it. Browse to `/opt/IBM/WebSphere/Appserver/java/jre/bin` and select **javaws**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

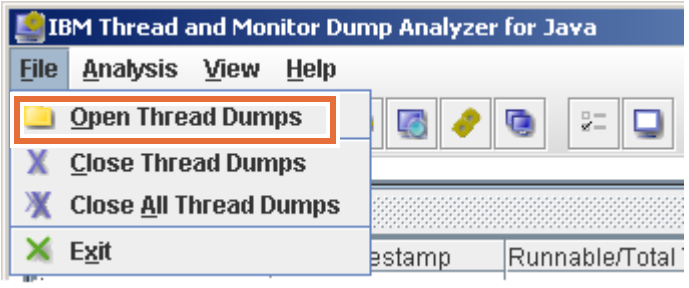
__ h. On the warning window, click **Run**.



__ i. The TMDA Desktop tool starts in its own window.

__ 5. Use the Thread and Monitor Dump Analyzer tool to examine and compare the javacore dumps.

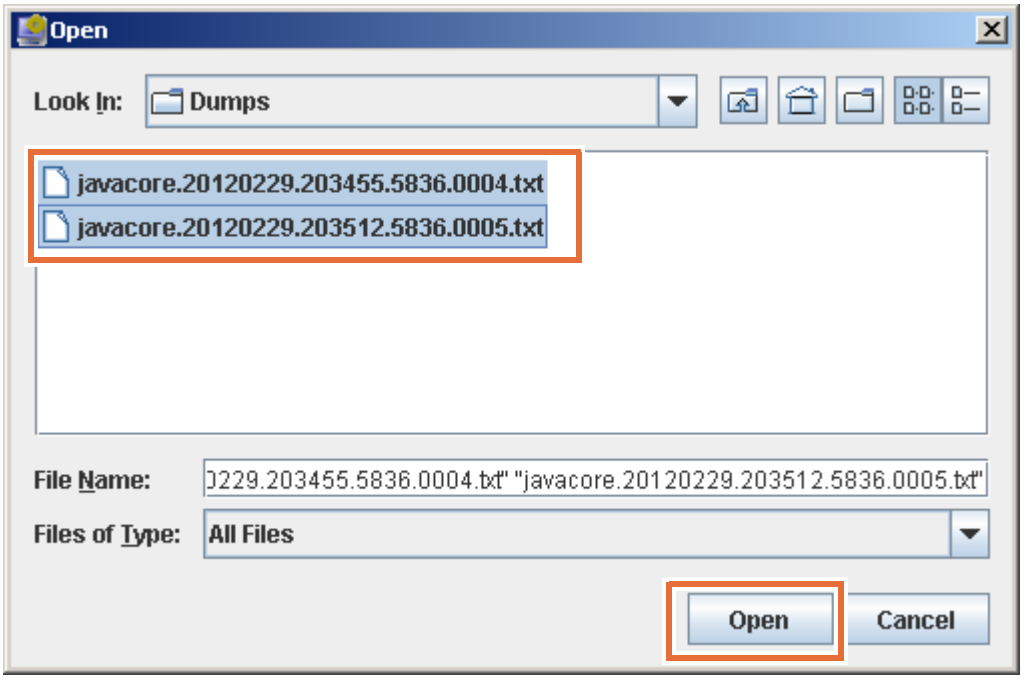
__ a. When the tool opens click **File > Open Thread Dumps**.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

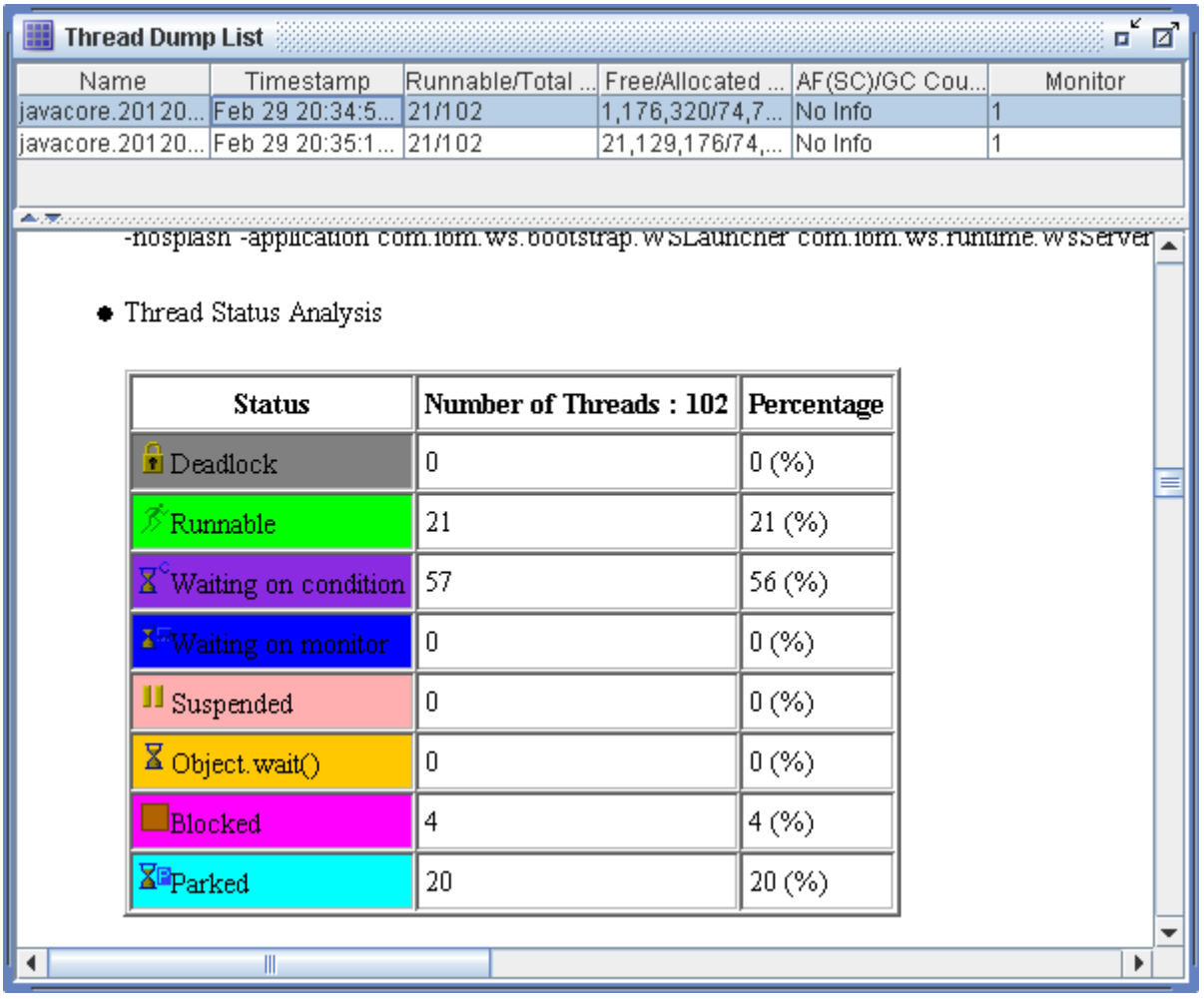
__ b. Browse to `/usr/Student/Hang/dumps` and use the Ctrl key to select both javacore files. Notice that your sequence numbers might be different.



__ c. Click **Open**. Wait a few seconds until both javacore files are analyzed.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. The TMDA tool enables you to examine each javacore individually or to compare threads between two or more javacore dumps. Select the first javacore and scroll down to see the **Threads Status Analysis**.



__ e. Remember that a javacore is a snapshot of the threads and their status at the time they were dumped. You might see 3 or more (4 if you are quick enough) **Blocked** threads as a result of the hang scenario you created with the BadApp. Recall that the five browsers were “hung” for several seconds when you generated this memory dump.

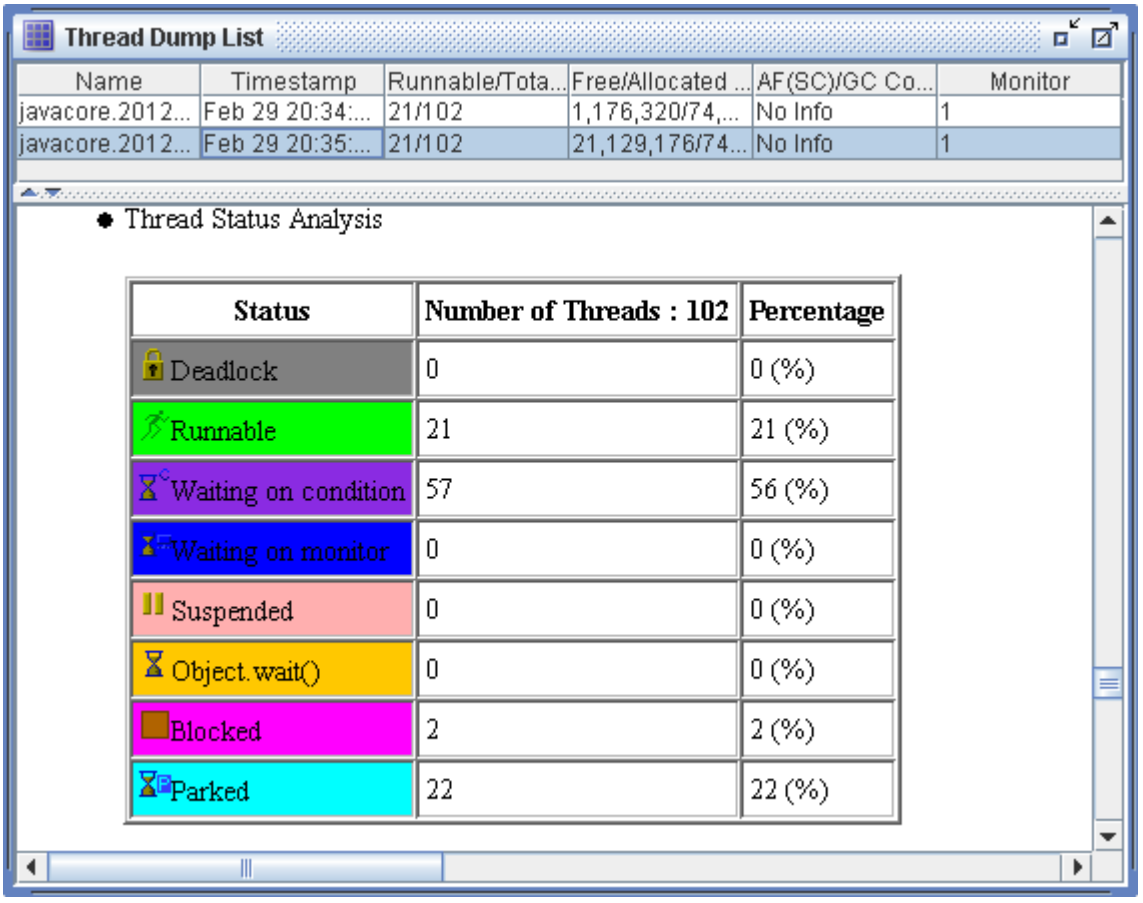
Note

If you are unable to capture the two javacore files in the previous activity, you can use the ones in the folder `/usr/Student/Hang/Threaddumps`.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

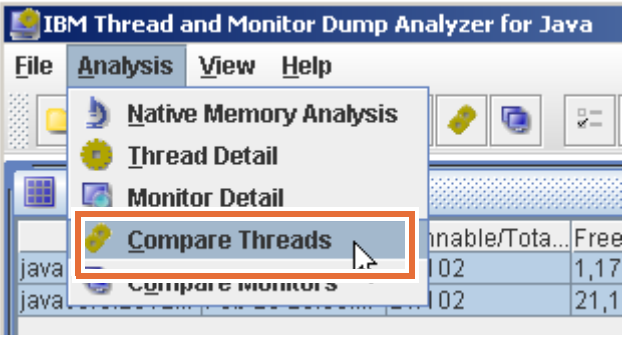
__ f. Select the second (later) javacore and compare its **Thread Status Analysis.**



__ g. You might notice 2 or fewer **Blocked** threads, indicating that between the first and the second thread dumps, some of the threads completed.

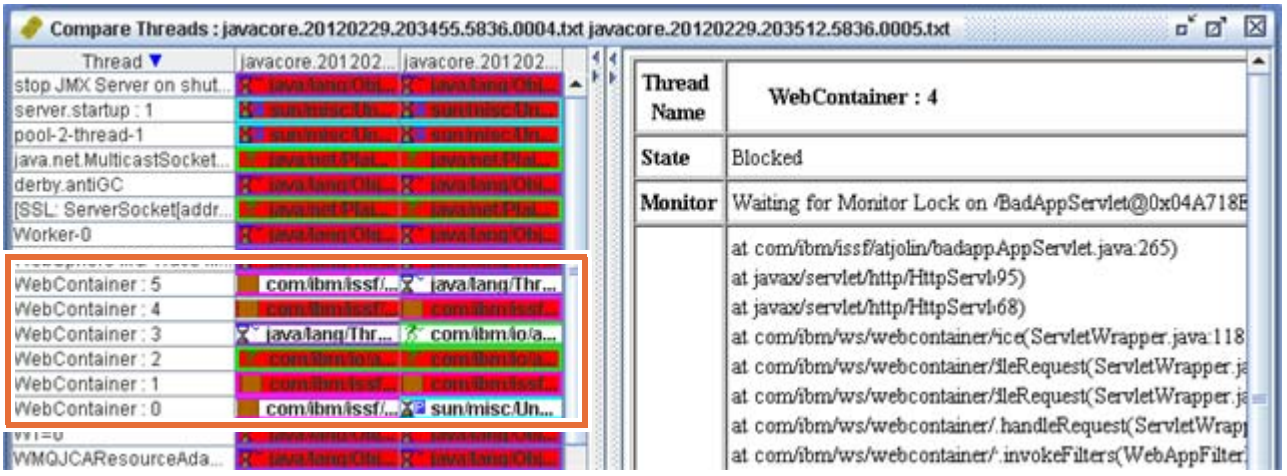
__ 6. Next, create a comparative analysis of the two javacore files.

__ a. Highlight both javacore files by using the Ctrl key in the Thread Dump List, and click **Analysis > Compare Threads**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. When the comparative analysis is displayed, click the top of the **Thread** column to sort the threads in reverse order. You want to see the WebContainer threads.



__ c. You see at least 5 WebContainer threads. The next two columns show the status of the threads from the javacore files side-by-side. Clicking the thread in columns two and three causes details to be displayed. The screen capture shows that the thread WebContainer 4 was selected in the Threads column.

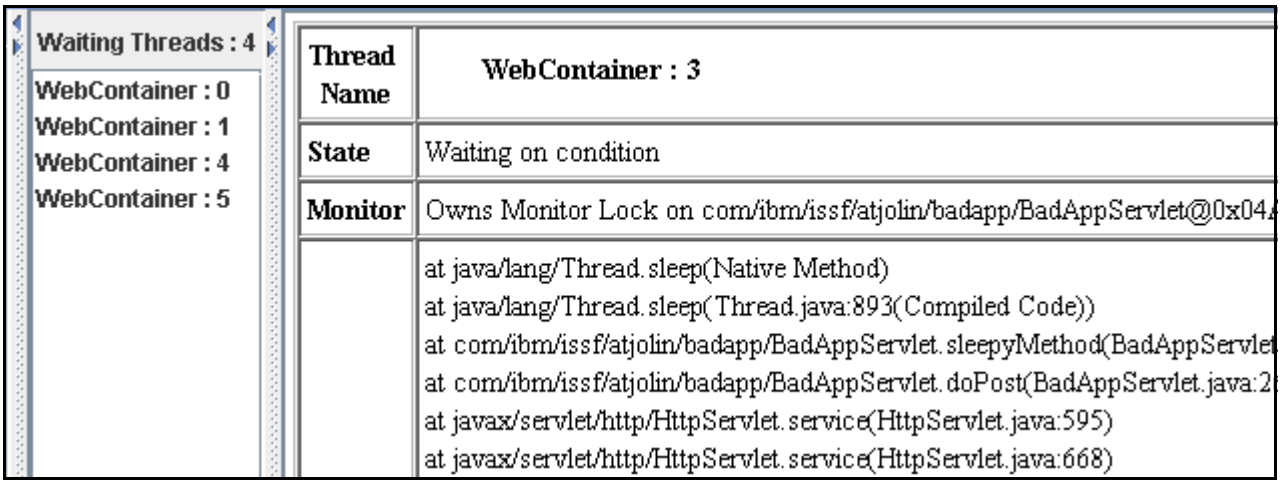
Thread Name	**WebContainer : 4**
State	Blocked
Monitor	Waiting for Monitor Lock on com/ibm/issf/atjolin/badapp/BadAppServlet@0x04A718E8
Java Stack	at com/ibm/issf/atjolin/badapp/BadAppServlet.doPost(BadAppServlet.java:265) at javax/servlet/http/HttpServlet.service(HttpServlet.java:595) at javax/servlet/http/HttpServlet.service(HttpServlet.java:668) at com/ibm/ws/webcontainer/servlet/ServletWrapper.service(ServletWrapper.java:1188) at com/ibm/ws/webcontainer/servlet/ServletWrapper.handleRequest(ServletWrapper.java:763) at com/ibm/ws/webcontainer/servlet/ServletWrapper.handleRequest(ServletWrapper.java:454)

__ d. Click each WebContainer thread and determine whether it is blocked. You notice that the blocked threads in the first memory dump were all in the `doPost` method of the BadAppServlet. Each of the blocked threads is "`Waiting for Monitor Lock on com/ibm/issf/atjolin/badapp/BadAppServlet`". Try to determine which thread has

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

the Monitor Lock for which the others are waiting. In this example, it was the **WebContainer: 3** thread.



__ e. Notice that you can open the Waiting Threads column and see which threads are waiting on the **WebContainer: 3** thread, which owns the Monitor Lock.

__ f. Observe the stack trace for the thread that owns the lock. Reading the stack trace from the bottom up after it invokes the `doPost` method and gets the lock, it goes to sleep, `Thread.sleep(Native Method)`!

__ g. After examining all threads, close TDMA by clicking **File > Exit** and **Yes**.

Information

Explanation of the BadApp behavior

In this hung thread scenario, the handling of requests was greatly slowed down. The cause was that a fairly common operation locked a particular resource for a fairly long amount of time. For any performance problem, the degree of slowdown depends on the actual value of “fairly” for a specific application. For BadApp, the operation is called for every request in Mode 2, and the lock is held for 10 seconds. In a real application, under a real user load, 1/10 second or even 1 ms would be a long time to hold a lock. The developer used 10 seconds so that you could see a slowdown while manually sending five requests from five browser windows.

For BadApp, a common form of lock contention, the `synchronize` method, was used. In this case, the resource that was locked (the HttpServlet), was the Java object whose method was called.

It is common that an application must execute some code in a serial way, meaning that only one thread can execute this code at any time. Updating a shared static variable is one example that is common, and writing to a shared log file is another. The Java `synchronize` mechanism was created explicitly to handle such situations where they cannot be avoided through other means. In well-written code, there usually is not a problem when using `synchronize`. However, a novice programmer might synchronize far more code than is needed, often synchronizing an entire method when only one or two lines of code are critical. In the scenario examined in this exercise, BadApp exhibited this flaw.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 7. Stop the IBM Support Assistant.

__ a. Close the IBM Support Assistant web browser.

__ b. From your system desktop, click **Applications**.

__ c. Select **IBM Support Assistant > Stop IBM Support Assistant 5**.

End of exercise

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Exercise review and wrap-up

In this exercise, you detected and diagnosed server hangs.

To diagnose hung threads, you configured the thread monitor of a server by using the hang detection facility. You also installed and ran a test application that was specially written to cause threads to hang and viewed the corresponding thread monitor output. Then, you created a javacore file (thread dump) from the running server. Next, you were able to use the Thread and Monitor Dump Analyzer tool to view an existing javacore file that contained deadlocks and read the analysis that the tool performed.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 5. Troubleshooting crashes

What this exercise is about

In this exercise, you learn how to troubleshoot an application server crash.

What you should be able to do

At the end of this exercise, you should be able to:

- Analyze a javacore file for a crash
- Use various methods to trigger a system core dump
- Use the IBM Monitoring and Diagnostic Tools for Java - Interactive Diagnostic Data Explorer to analyze system core files

Introduction

Another problem that can occur is an application server crash. Crashes occur when an invalid operation is done.

Crashes can be categorized into three main types:

- System failures that result from some problem in the Java code, such as OutOfMemory, StackOverflow, and other errors.
- Crashes that result from an internal error or defect in the JVM implementation. These defects can be fixed by applying a JDK patch.
- System failures that result from an internal error in some external third-party JNI library that customers use and load in their JVM.

For crashes that result from OutOfMemory and StackOverflow, the JVM usually gets a chance to do some cleanup before exiting. In particular, it usually produces a javacore file, which can then be used to help determine the nature of the crash.

Requirements

Two tools are required:

- IBM Monitoring and Diagnostic Tools for Java – Memory Analyzer
- Interactive Diagnostic Data Explorer (IDDE)

Both tools are used. These tools are preinstalled and are accessed from the IBM Support Assistant and the IBM Assembly and Deploy Tools.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Start the servers

__ 1. Ensure that the deployment manager is started.

__ a. From a command prompt, change directory to:

`<profiles_root>/Dmgr/bin`

__ b. Enter the command:

```
./serverStatus.sh dmgr -user wasadmin -password web1sphere
```

__ c. If the dmgr is stopped, start it with the command: `./startManager.sh`

__ 2. Ensure that the node agent and server1 are started.

__ a. From a command prompt, change directory to:

`<profiles_root>/profile1/bin`

__ b. Enter the command: `./serverStatus.sh -all`

__ c. If the node agent is stopped, start it with command: `./startNode.sh`

__ d. If server1 is stopped, start it with command: `./startServer.sh server1`

Section 2: Analyze a javacore to troubleshoot a server crash

In this part, you analyze an existing javacore file that was generated as a result of a Java process crash. You can use any text editor to view javacore files. You use WordPad in this exercise.

__ 1. Open the existing javacore file.

__ a. From a Terminal window, navigate to: `/usr/Student/Crash`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Use gedit to open an example javacore file by typing:

```
gedit javacore_JIT.txt
NULL
------------------------------------------------------------------------
0SECTION       TITLE subcomponent dump routine
NULL           ================================
1TISIGINFO     Dump Event "gpf" (00002000) received
1TIDATETIME    Date:                 2012/02/29 at 08:53:33
1TIFILENAME    Javacore filename:
/opt/IBM/WebSphere/ProcServer7/profiles/UTM01/javacore.20120229.085318
.11206874.0003.txt
1TIREQFLAGS    Request Flags: 0x81 (exclusive+preempt)
1TIPREPSTATE   Prep State: 0x0
1TIPREPINFO    Exclusive VM access not taken: data may not be consistent across
javacore sections
NULL
------------------------------------------------------------------------
0SECTION       GPINFO subcomponent dump routine
NULL           ================================
2XHOSLEVEL     OS Level         : AIX 6.1
2XHCPUS        Processors -
3XHCPUARCH       Architecture   : ppc64
3XHNUMCPUS       How Many       : 28
3XHNUMASUP       NUMA is either not supported or has been disabled by user
NULL
1XHEXCPCODE    J9Generic_Signal_Number: 00000010
1XHEXCPCODE    Signal_Number: 00000004
1XHEXCPCODE    Error_Value: 00000000
1XHEXCPCODE    Signal_Code: 0000001E
1XHEXCPCODE    Handler1: 09001000A10A38A8
1XHEXCPCODE    Handler2: 09001000A109AE58
NULL
NULL
```

__ c. First, notice the `1TISIGINFO` line. The Dump Event is `"gpf"`, which stands for general protection fault. The `GPINFO` sections show that the OS is AIX 6.1 and the processor is PPC64, a 64-bit system.

Information

Other signals such as signal 3 (SIGQUIT) and signal 21 (SIGBREAK on Windows) indicate that the javacore is created with an explicit command. You saw this signal in the hang detection exercise. In those cases, you would not see crash information that pinpoints the module or library that caused the crash.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Search for the `1XHEXCPMODULE` line. One or more of these lines show where the crash occurred.

```
1XHEXCPMODULE  Compiling method:
com/ccure/local/ab/charter/StatsPollerImpl.pollStats.(Lcommonj/sdo/DataObj
ect;)V
```

From the `1XHEXCPMODULE` line, it is possible that the JIT compiler compiled native code.

__ e. Next, scroll down to the `ENVINFO` section.

```
NULL
------------------------------------------------------------------------
0SECTION       ENVINFO subcomponent dump routine
NULL           ================================
1CIJAVAVERSION JRE 1.6.0 IBM J9 2.4 AIX ppc64-64 build jvmap6460sr7-20100219_54049
1CIVMVERSION   VM build 20100219_054049
1CIJITVERSION  JIT enabled, AOT enabled - r9_20091123_13891
1CIGCVERSION   GC - 20100216_AA_CMPRSS
1CIRUNNINGAS   Running as a standalone JVM
1CICMDLINE     /opt/IBM/WebSphere/ProcServer7/java/bin/java -Declipse.sec
1CIUSERARGS    UserArgs:
2CIUSERARG               -Xjcl:jclscar_24
2CIUSERARG               -Dcom.ibm.oti.vm.bootstrap.library.path=/opt/IBM/Web
```

__ f. Notice that the Java version and process command line are presented. Also, there is a complete list of command-line arguments for your inspection. It is important to know that JIT and AOT are enabled. This information is necessary for verifying the runtime environment of the server.

__ g. Search for the **Threads** section to see the **Current thread**.

```
NULL
------------------------------------------------------------------------
0SECTION       THREADS subcomponent dump routine
NULL           ================================
NULL
1XMCURTHDINFO  Current thread
NULL           ----------------------
3XMTHREADINFO      "JIT Compilation Thread" J9VMThread:0x0000000030002A00,
j9thread_t:0x0000000110155020, j
ava/lang/Thread:0x0000000040035DF0, state:R, prio=10
3XMTHREADINFO1           (native thread ID:0x105029D, native priority:0xB, native
policy:UNKNOWN)
3XMTHREADINFO3           No Java callstack associated with this thread
3XMTHREADINFO3           Native callstack:
4XENATIVESTACK               (0x0900000005C07F60 [libj9jit24.so+0x448f60])
4XENATIVESTACK               (0x0900000005BFF2DC [libj9jit24.so+0x4402dc])
... <callstack truncated> ...
```

__ h. Notice that the current thread is `"JIT Compilation Thread"` indicating that the crash resulted from a problem (gpf) during the JIT compilation process.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ i. Close the `javacore_JIT.txt` file.

__ 2. **Optional**: If you have time, try analyzing the other javacore in the `/usr/Student/Crash` directory named `javacore_CompRefs.txt`. Look at the `SIGINFO` and `HEXCPMODULE` lines to determine the cause of the crash. Look at the `Threads` section to discover what thread was running at the time of the crash.

Section 3: Use the Memory Analyzer to examine a system core file

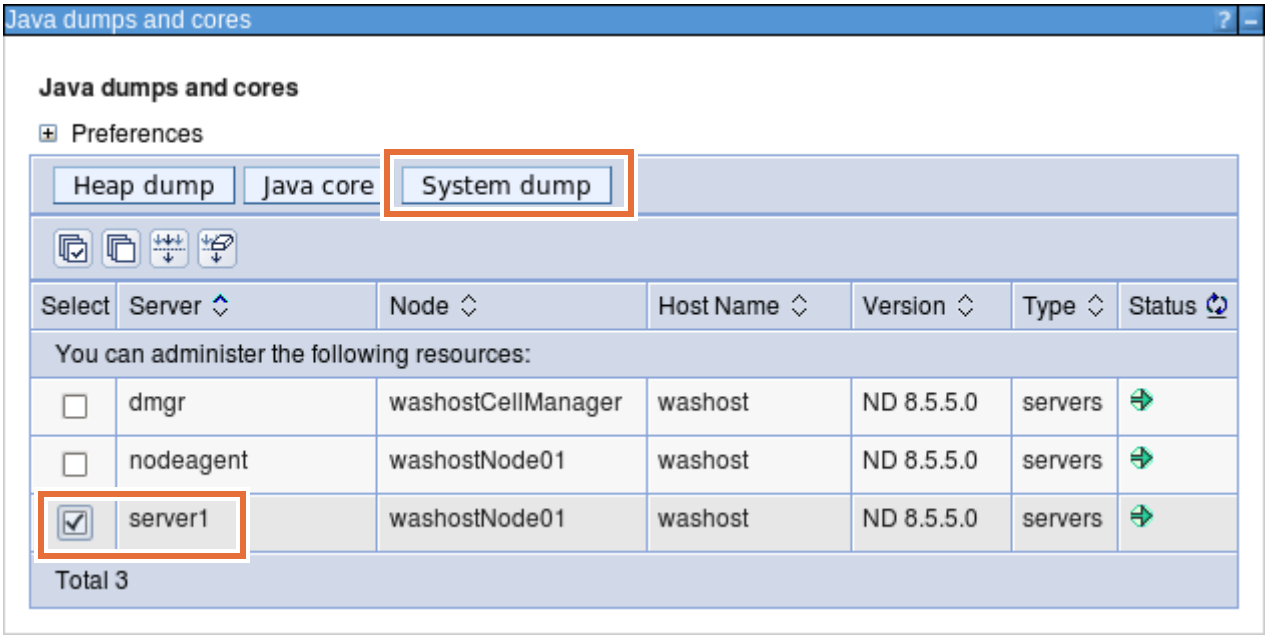
The IBM Monitoring and Diagnostic Tools for Java – Memory Analyzer uses analyzers to interrogate a formatted system dump (each analyzer asks the dump a specific question) and links them together with a script to produce a concise report of the analysis. Typical problems analyzed might be:

- Out of memory
- Deadlock detected
- VM terminated due to signal (internal or middleware Java application error)

The objective is to diagnose the correct category of problem. Then, either provide a complete diagnosis of the problem or give some information that is designed to aid diagnosis together with a suggested next course of action.

__ 1. Use the administrative console to trigger a system core dump.

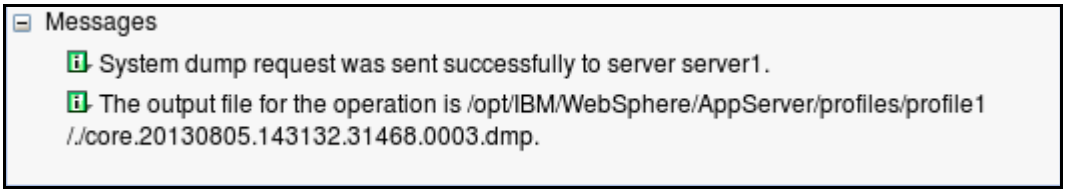
__ a. From the administrative console, select **Troubleshooting > Java dumps and cores**.



__ b. Select **server1**, and click **System dump**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. You see a console message when the request completes.



__ d. Move the core file from `/opt/IBM/WebSphere/AppServer/profiles/profile1` to `/usr/Student/Crash/SystemDump` and rename it: `core.wasPD.dmp`

Information

You can also use wsadmin and Jython commands to trigger a system core dump.

```
jvm=AdminControl.completeObjectName('type=JVM,process=server1,*')
AdminControl.invoke(jvm, 'generateSystemDump')
```

__ 2. In order for some tools like the Dump Analyzer to analyze the core dump file, the jextract tool must first process it. Run the jextract tool.

__ a. From a Terminal window, navigate to:

`/opt/IBM/WebSphere/AppServer/java_1.7_64/jre/bin/`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Run the following command:

```
./jextract /usr/Student/Crash/SystemDump/core.wasPD.dmp
```

```
root@washost:/opt/IBM/WebSphere/AppServer/java_1.7_64/jre/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./jextract /usr/Student/Crash/SystemDump/core.wasPD.dmp
Loading dump file...
Read memory image from /usr/Student/Crash/SystemDump/core.wasPD.dmp
XML production is switched off by default, specify -xml to override
Creating archive file: /usr/Student/Crash/SystemDump/core.wasPD.dmp.zip
Adding "/usr/Student/Crash/SystemDump/core.wasPD.dmp"
Adding "/lib64/libnss_files.so.2"
Adding "libj9zlib26.so" (found at "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd
64/compressedrefs/libj9zlib26.so")
Adding "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd64/compressedrefs/libvmi.so
"
Adding "libj9ute26.so" (found at "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd6
4/compressedrefs/libj9ute26.so")
Adding "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd64/compressedrefs/libhythr.
so"
Warning:  Could not find file "libWs60ProcessManagement.so.1" for inclusion in a
rchive
Adding "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd64/libjava.so"
Adding "/opt/IBM/WebSphere/AppServer/java/bin/java"
Adding "libjclscar_26.so" (found at "/opt/IBM/WebSphere/AppServer/java/jre/lib/a
md64/compressedrefs/libjclscar_26.so")
Adding "libj9hookable26.so" (found at "/opt/IBM/WebSphere/AppServer/java/jre/lib
/amd64/compressedrefs/libj9hookable26.so")
Adding "/opt/IBM/WebSphere/AppServer/java/jre/lib/amd64/j9vm/libjvm.so"
```

__ c. Verify that the command completes with the following message: `jextract complete`. Verify that a compressed file, `core.wasPD.dmp.zip`, is written to the same directory as the `core.wasPD.dmp` file.

__ 3. Review the system core file with the Memory Analyzer. Make sure that IBM Support Assistant is started. You started it in the previous sections, but if you stopped it, follow these steps:

__ a. Start the IBM Support Assistant by clicking **Applications** on the toolbar. Select **Start IBM Support Assistant 5**.

__ b. A terminal window pops up. Wait until you see a message that says the IBM Support Assistant is started. Do not close this window until you are finished with the IBM Support Assistant.

__ 4. Launch the **Memory Analyzer [Desktop]** from the **IBM Support Assistant**.

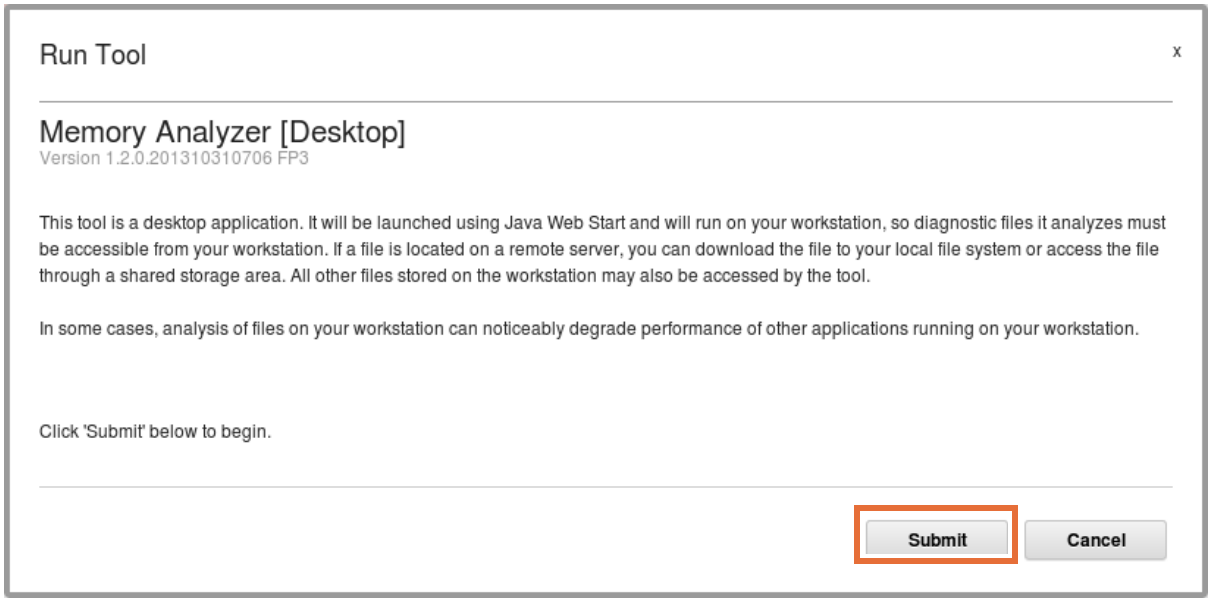
__ a. Open a web browser, and enter the web address:

http://washost:10911/isa5

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. You can launch the **Memory Analyzer [Desktop]** from the **Tools** tab of the IBM support Assistant.

__ c. Click **Submit**.



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

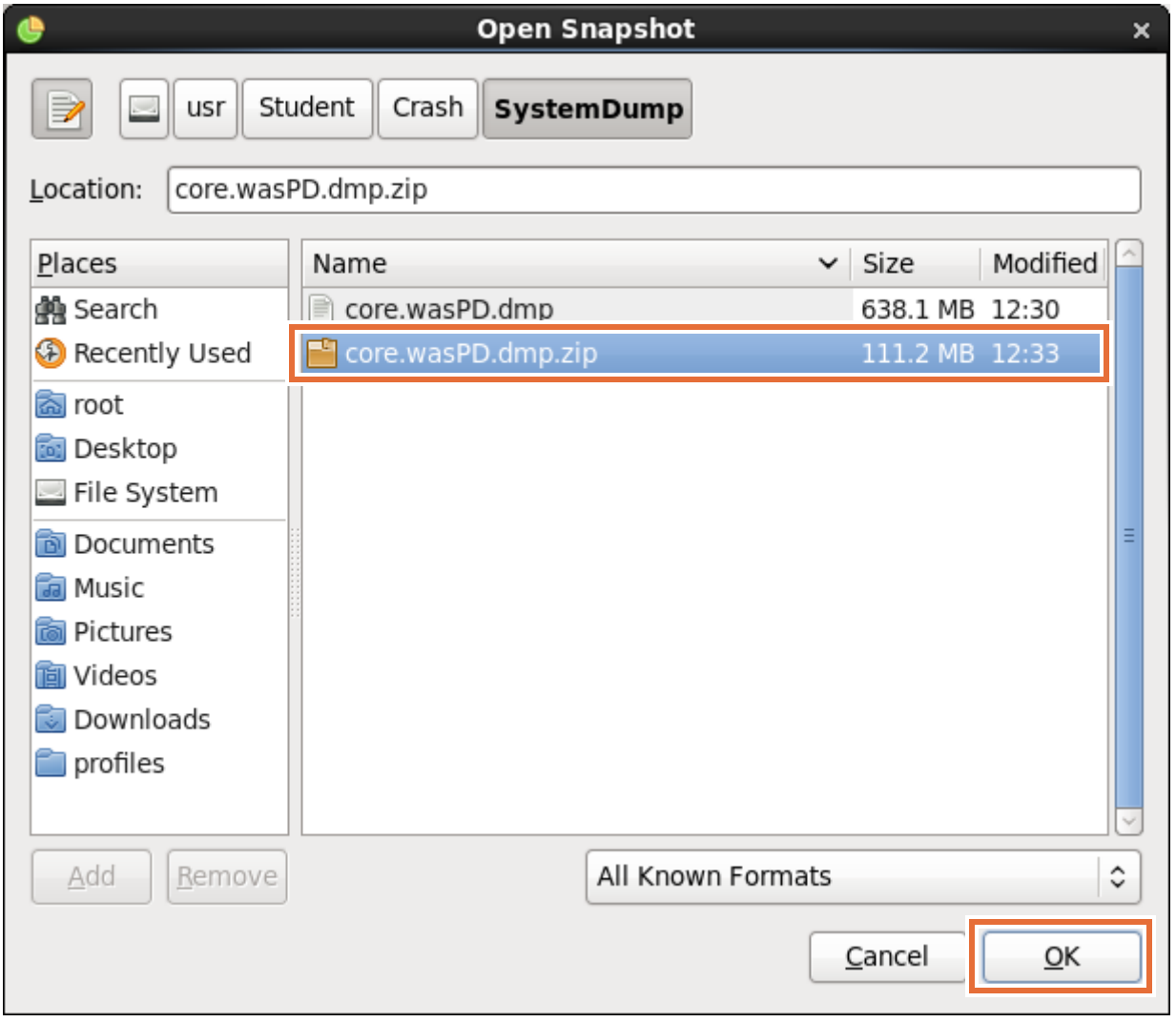
__ d. Check "**Always trust content from this publisher**" and click **Yes** if prompted.

__ e. Check "**Always trust content from this publisher**" and click **Run** if prompted with a warning about unrestricted access.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 5. Load and analyze the system core file with the Memory Analyzer.

__ a. From the File Menu select Open Heap Dump... and navigate to `/usr/Student/Crash/SystemDump` and select `core.wasPD.dmp.zip`.

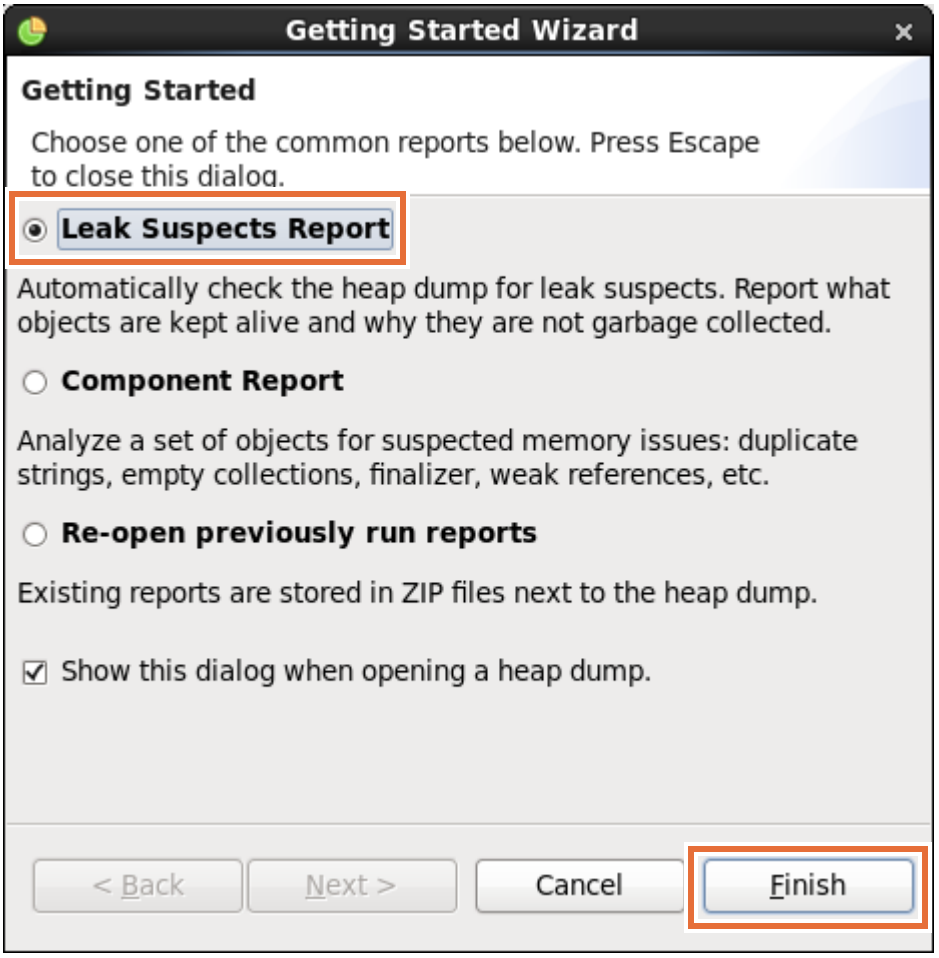


__ b. The **Memory Analyzer** will take a few minutes to process this file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

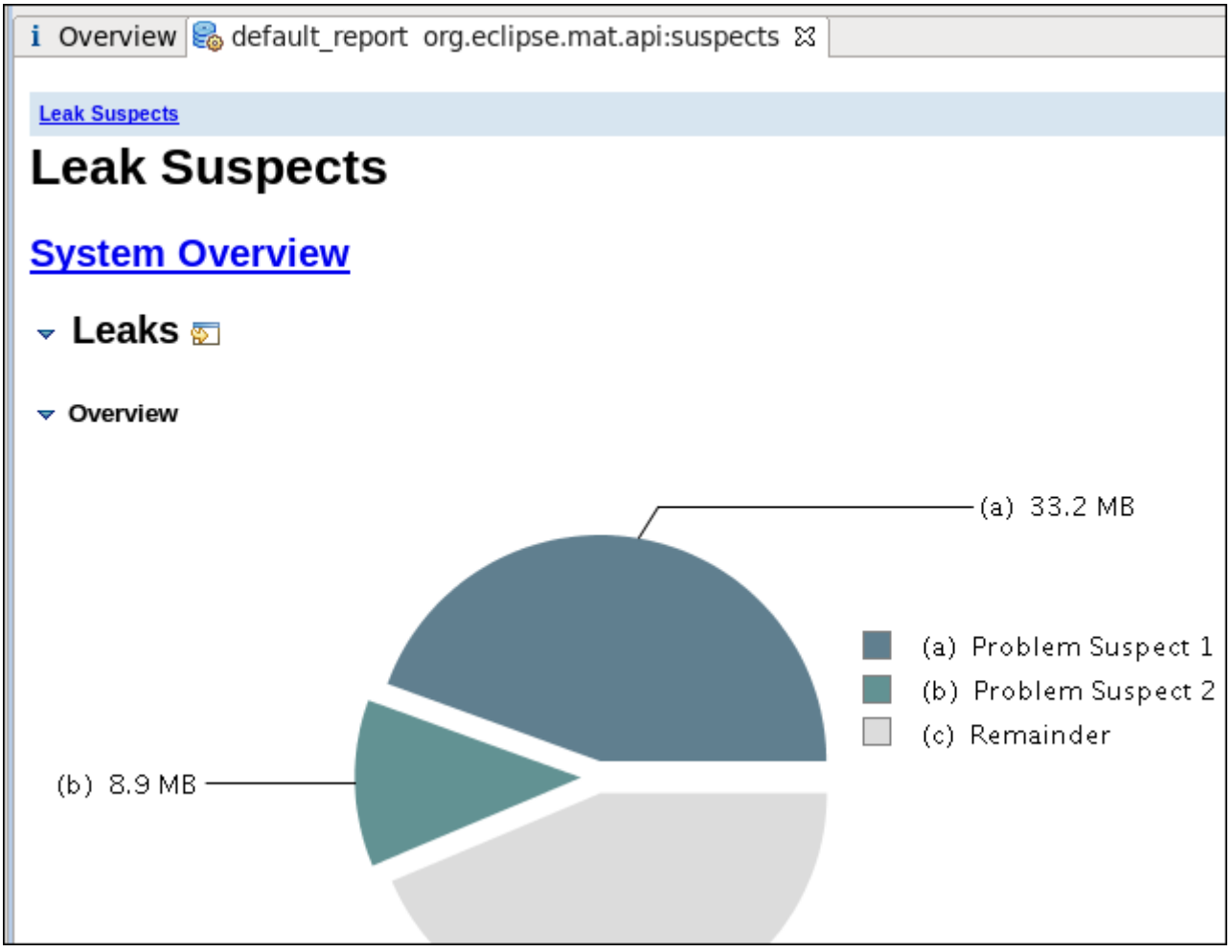
__ c. Select **Leak Suspects Report** and click **Finish**.



__ d. The **Memory Analyzer** will take a few minutes to continue processing the file.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. The default_report pane will show a pie chart of the **Leak Suspects**.



__ f. Scroll down to see the details for the suspicious objects.

__ g. Take a look at some of the other panes in the **Memory Analyzer** tool. When you are done close the tool.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Note

The Memory Analyzer tool will save a new copy of the system dump that includes the analysis. Take a look at the directory /usr/Student/Crash/SystemDump to see a new compressed file.

`core.wasPD.dmp`

`core.wasPD.dmp_Leak_Suspects.zip`

`core.wasPD.dmp.zip`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 4: Use the Interactive Diagnostic Data Explorer (IDDE) to analyze system core files

Interactive Diagnostic Data Explorer (IDDE) is the strategic tool for allowing interactive analysis of JVM problems by using postmortem artifacts such as core files or javacore files. It is a lightweight tool that helps you quickly get information from the artifact you are investigating when you are not sure what the problem is and you want avoid launching resource-intensive analysis.

Important

If you are using the IBM Remote Lab Platform (IRLP) environment, disregard this note.

Note: There is a problem running IDDE tool with a web browser-based VNC Viewer.

In order to issue commands to the IDDE tool you must be able to access it in the IBM Support Assistant with a remote web browser or standalone Real VNC Viewer

1. Remote web browser. Use Firefox on your desktop and enter the URL

   http://<your_image_IP_address>:10911/isa5

2. Real VNC Viewer. Connect to you lab image with standalone Real VNC Viewer and use the Firefox browser on the image

__ 1. Make sure that IBM Support Assistant is started. You started it in the previous sections, but if you stopped it, follow these steps:

__ a. Start the IBM Support Assistant by clicking **Applications** on the toolbar. Select **Start IBM Support Assistant 5**.

__ b. A terminal window pops up. Wait until you see a message that says the IBM Support Assistant is started. Do not close this window until you are finished with the IBM Support Assistant.

__ 2. Launch the **Interactive Diagnostic Data Explorer (IDDE) [Desktop]** from the **IBM Support Assistant**.

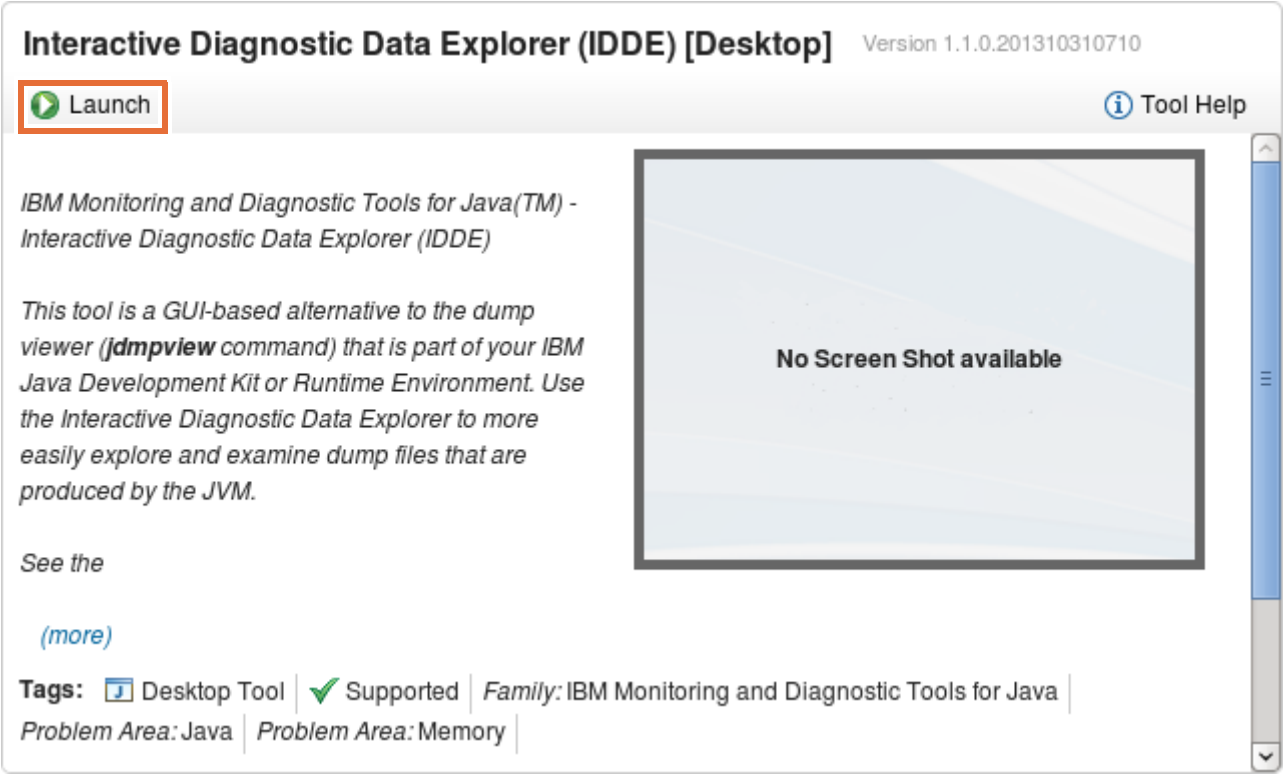
__ a. Open a web browser, and enter the web address:

http://washost:10911/isa5

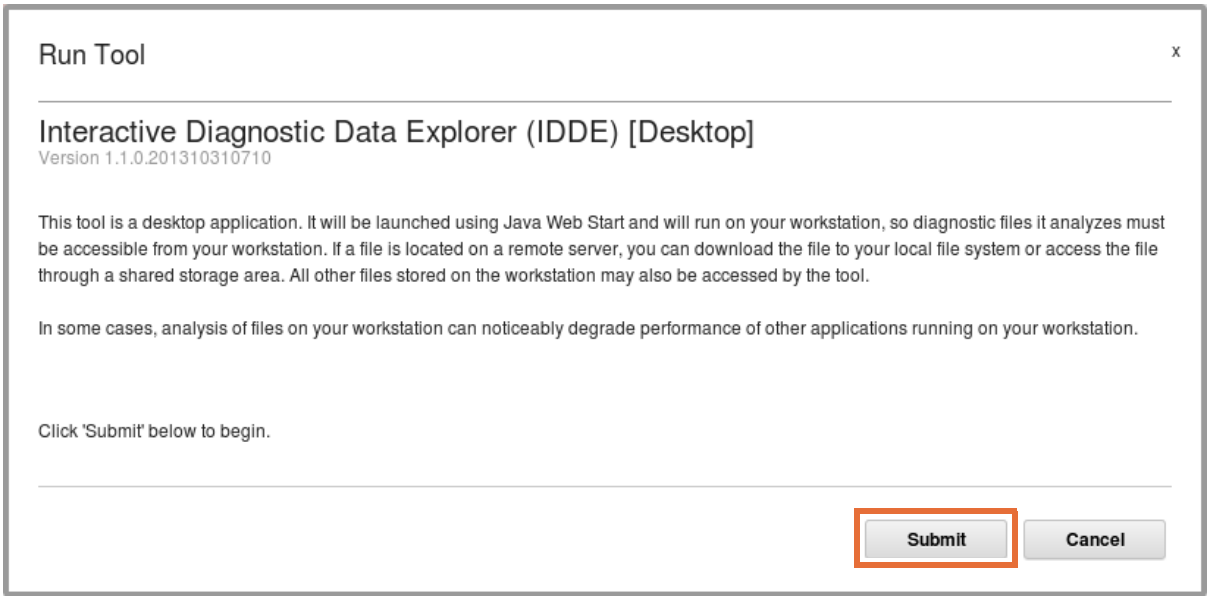
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. You can launch the **IDDE [Desktop]** from the **Tools** tab of the IBM support Assistant.



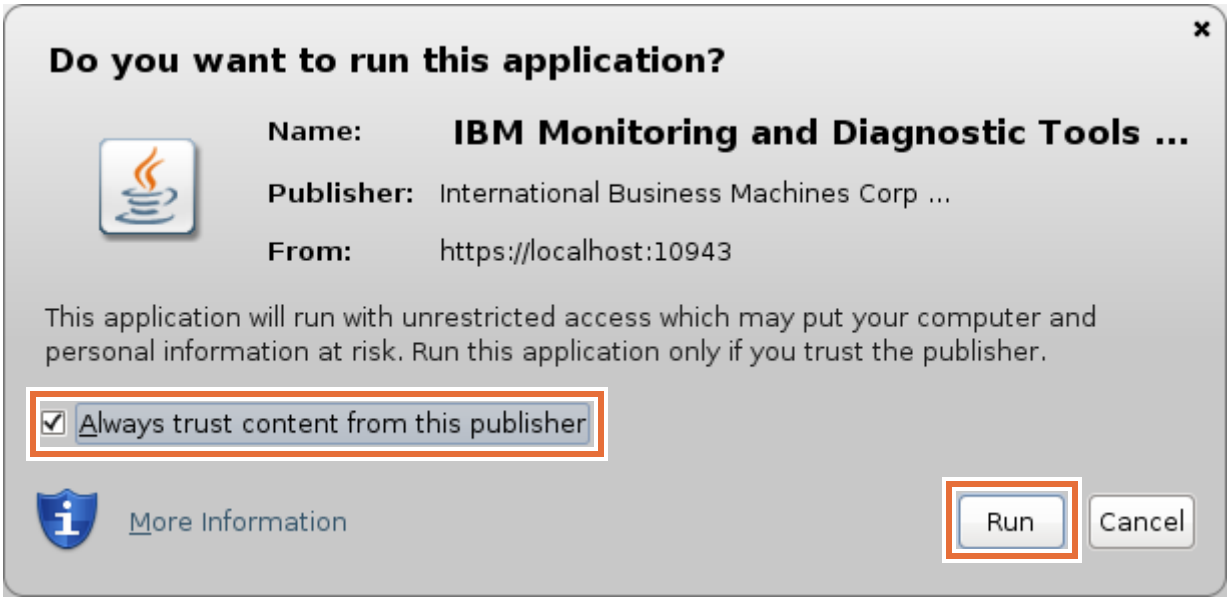
__ c. Click **Submit**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Check "**Always trust content from this publisher**" and click **Yes** if prompted.

__ e. Check "**Always trust content from this publisher**" and click **Run** if prompted with a warning about unrestricted access.

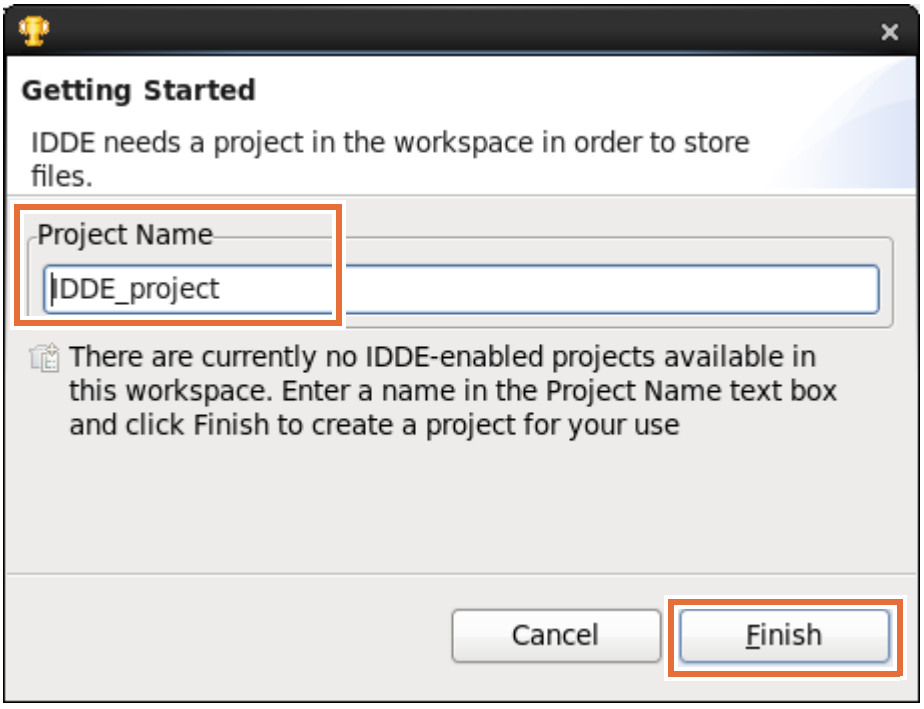


© Copyright IBM Corp. 2013

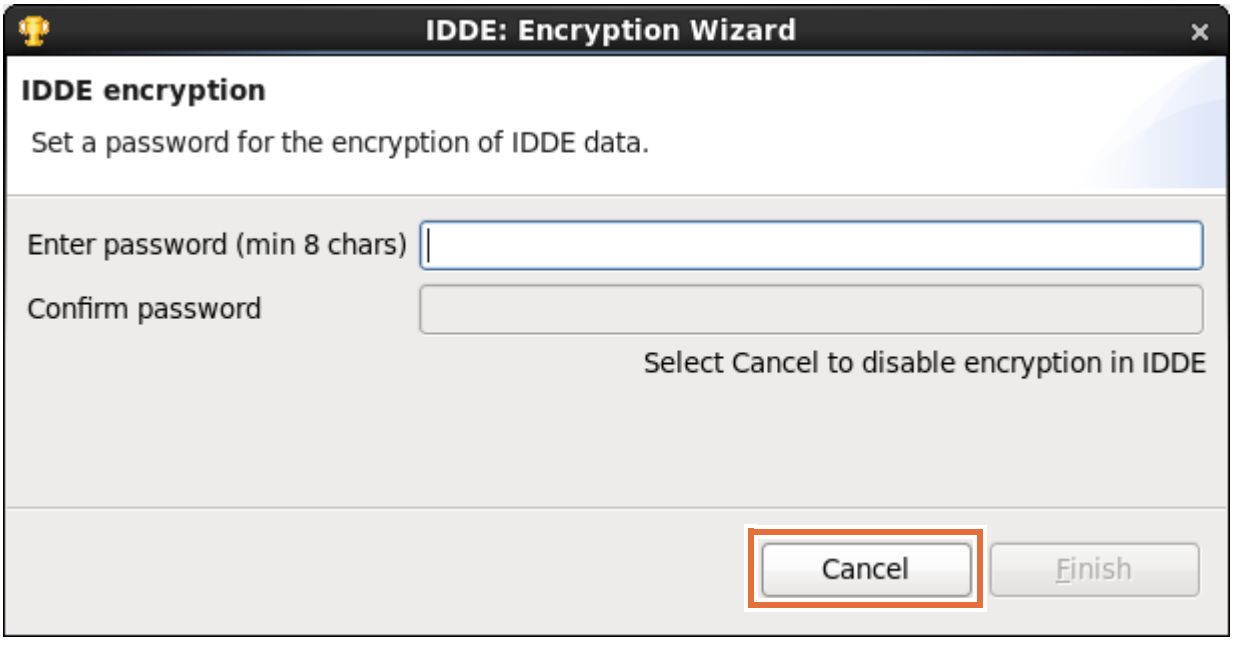
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 3. Create a project workspace.

__ a. Use the project name **IDDE_project** and click **Finish.**



__ b. Click Cancel to disable encryption of the IDDE workspace.



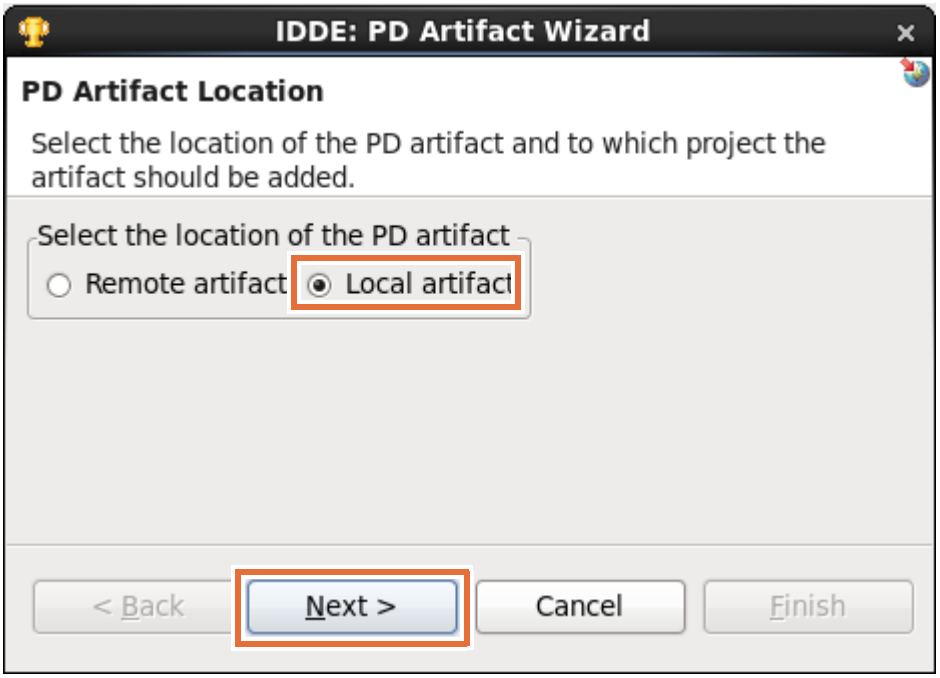
__ c. Click Yes to continue without encryption.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Note

Since you running in a test environment with no sensitive data, there is no reason to protect the data. When using IDDE with data that should not be publicly available, it is an excellent practice to encrypt the IDDE workspace.

__ d. Select **Local artifact** and click **Next**.

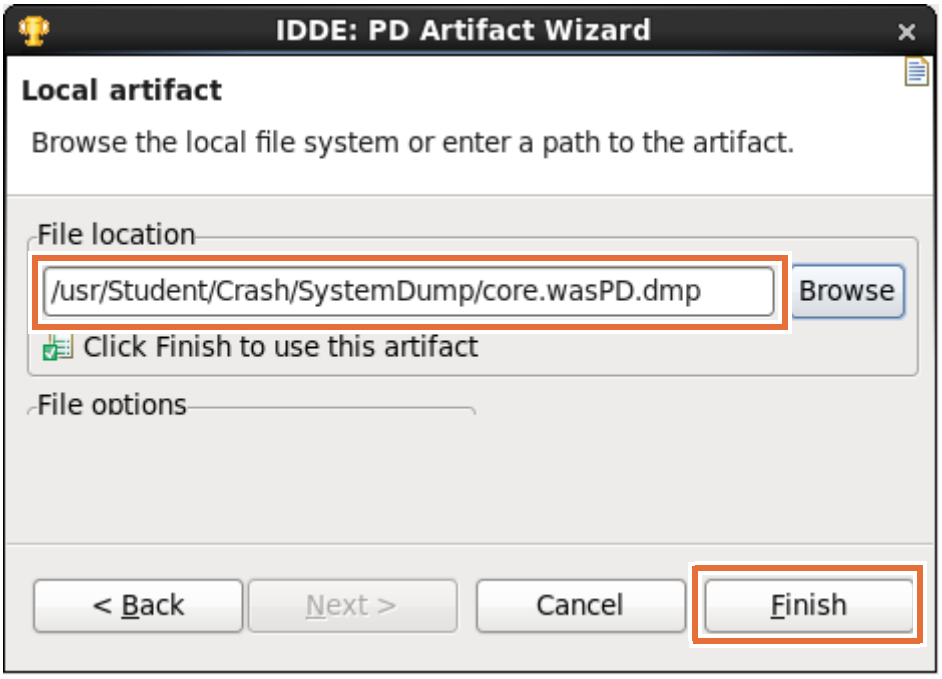


© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

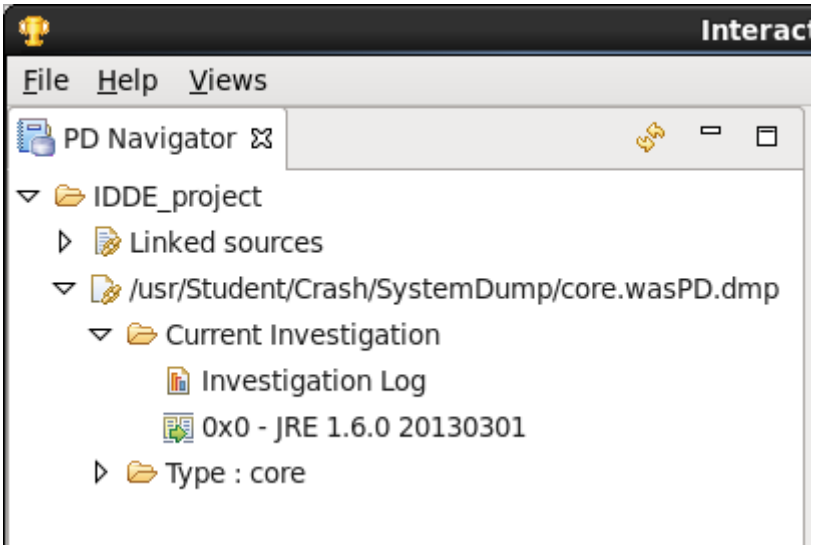
__ 4. Add a crash artifact.

__ a. Browse to the location of the `core.wasPD.dmp` file in the directory `/usr/Student/Crash/SystemDump`.



__ b. Click **Finish**.

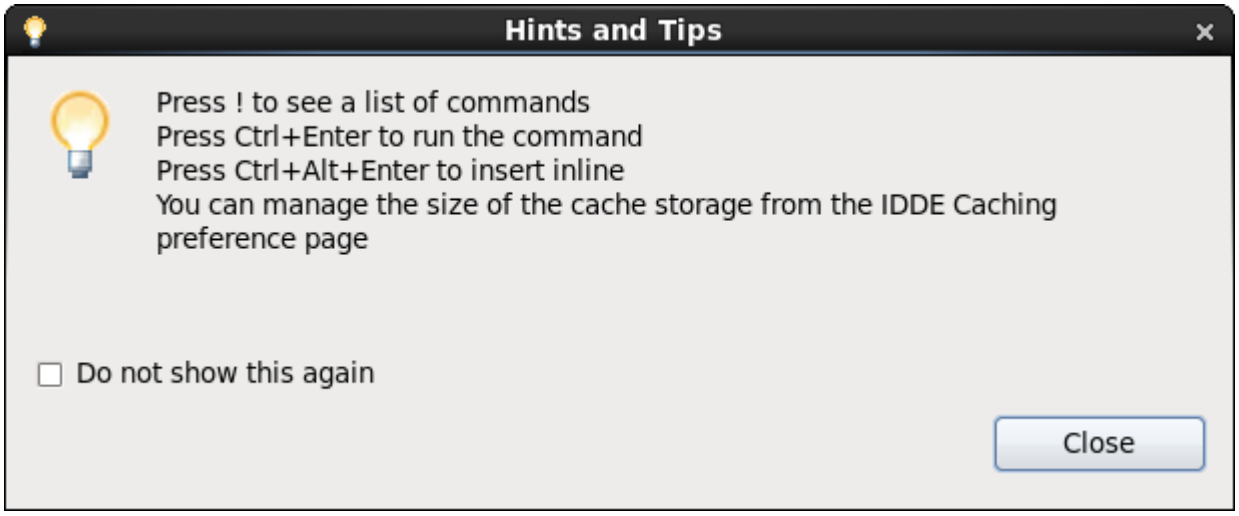
__ c. In the PD Navigator pane, expand the **IDDE_Project** folder, and double-click or expand **Current Investigation**. Analysis of the artifact begins, and the results are stored in the Current Investigation folder.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

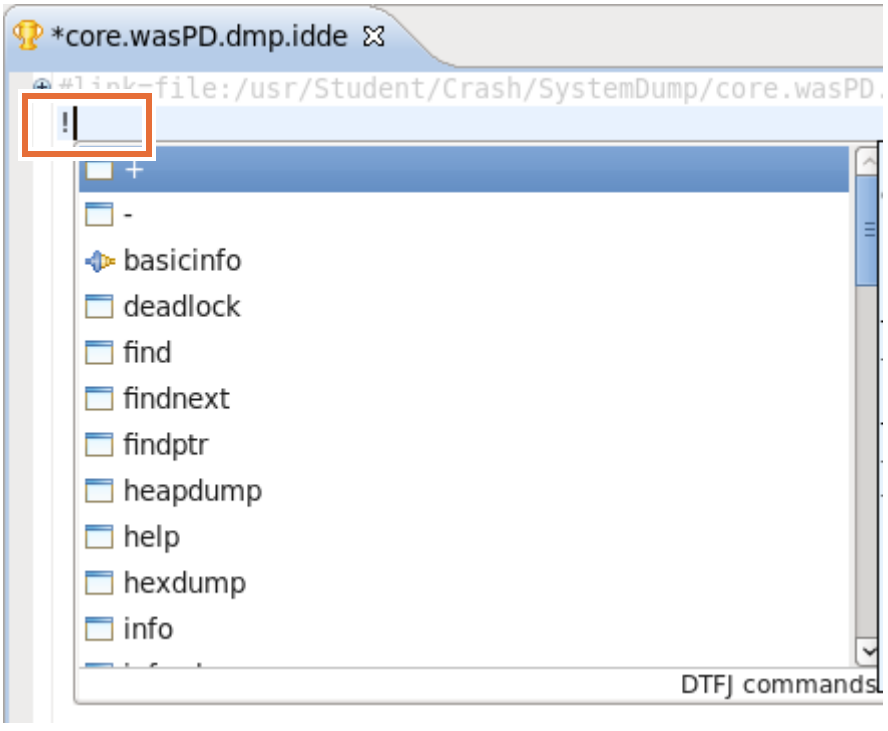
__ 5. Use commands to explore the contents of the system core dump.

__ a. Initially you see the Hints and Tips dialog.



__ b. Notice that typing "!" provides a list of commands. If you enter a command, press **Ctrl+Enter** to run the command. Click **Close**.

__ c. In the editor pane of IDDE, type "!" to see a help menu.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Near the top of the menu is a list of DTFJ commands. Highlight any of these commands to see a description of the command and its usage. Scroll down to the `info jitm` command, and highlight it to see its description.

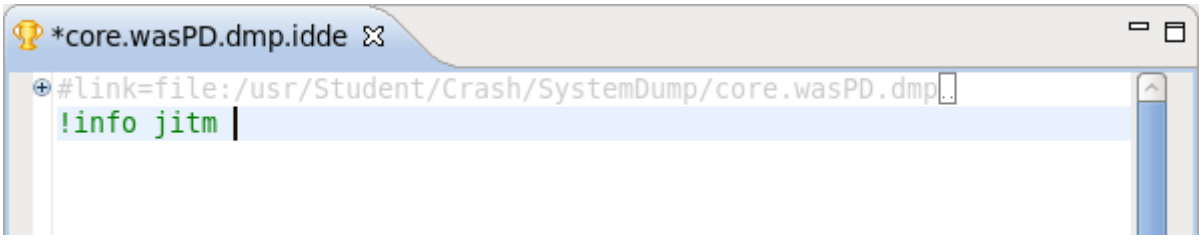
```
displays JIT'ed methods and their addresses

parameters: none

prints the following information about each JIT'ed method

  - method name and signature
  - method start address
  - method end address
```

__ e. If you suspect that a crash resulted from a JIT compile method, this information can be helpful. Double-click the `info jitm` command in the list. The command is entered in the editor.



__ f. Run the `info jitm` command by pressing **Ctrl+Enter**.

__ g. You see a list of methods with their start and stop addresses, name, and signature. Scroll through the list.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 6. Explore the `javathreads` command.

__ a. Type "`!`" in the editor as you did earlier to display the list of commands. Scroll down to the `javathreads` command, and run it.

```
*core.wasPD.dmp.idde

!javathreads {
Format is !javathreads [detail] | [list]
 eg !javathreads list
    To show an alphabetically sorted list of all the thread names
 eg !javathreads
    To show all the threads and java stacks
 or !javathreads detail
    To show all the threads and java stacks as well
     as compilation level and link to the class
To get the roots from the stacks of the threads use !javaheaproots
-----------------------------------------------------------------
ThreadName: AIO Timer Thread 1
To get more details of this particular thread:
     !javathread AIO Timer Thread 1
 ThreadStack:
  java/lang/Thread.sleep(JI)V
   java/lang/Thread.sleep(J)V 897
    com/ibm/io/async/Timer.run()V 185

ThreadName: Approximate Time Keeping Thread
To get more details of this particular thread:
   !javathread Approximate Time Keeping Thread
 ThreadStack:
  java/lang/Thread.sleep(JI)V
   java/lang/Thread.sleep(J)V 897
    com/ibm/ws/timeutils/QuickApproxTimeT$TimeThread.run()V 129
     java/lang/Thread.run()V 773
```

__ b. Scroll down and examine the list of threads. Notice that for each thread that is listed, a command is provided to get more details. An example is: `!javathread AIO Timer Thread 1`

You can copy and paste the command in the editor to run it. Or, you can place the cursor after the command, and press **Ctrl+Enter**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select any thread in the list, and run the command for more details. For example, you can select one of the WebContainer threads and examine the data.

```
*core.wasPD.dmp.idde
 !javathread WebContainer : 5 {
To get the roots from the stacks of all threads use !javaheaproots
----------------------------------------------------------------
ThreadName: WebContainer : 5
 com/ibm/ws/util/ThreadPool$Worker !javaobject 0x0000000003d0ed88
 Priority: 5 isDaemon=true State: STATE_ALIVE, STATE_PARKED,
    STATE_WAITING, STATE_WAITING_WITH_TIMEOUT
 ThreadStack:
  sun/misc/Unsafe.park(ZJ)V
  CompilationLevel=0 !javaclass sun/misc/Unsafe 0x00000000107b6900
   java/util/concurrent/locks/LockSupport.parkNanos(Ljava/lang/Object;J)
     <...Note: complete thread stack is truncated...>
```

__ 7. You can save your IDDE session, by clicking **Ctrl+S** or by clicking **File > Save**.

__ 8. Close the IDDE tool.

__ a. Click **File > Exit**.

Information

Resuming a saved IDDE session

If you save a session, you can resume it the next time that you start IDDE and run more commands to analyze the core dump.

In this exercise, you added the `core.wasPD.dmp` as an artifact, but it is also possible to add and analyze the jextracted file, `core.wasPD.dmp.zip`.

End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In this exercise, you examined javacores and system cores that result from server crashes. To investigate a server crash, you read the javacore file that was created when a test JVM failed. The provided javacore file was the result of a crash of a stand-alone Java application. Finally, you used the Interactive Diagnostic Data Explorer (IDDE) to analyze a system core file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 6. Troubleshooting an out-of-memory condition

What this exercise is about

This exercise explores some of the tools that can be used to detect and troubleshoot out-of-memory exceptions and memory leaks.

What you should be able to do

At the end of this exercise, you should be able to:

- Use Tivoli Performance Viewer to detect out-of-memory conditions
- Configure the lightweight memory leak detection mechanism
- Obtain verbose GC data and interpret it by using the Garbage Collection and Memory Visualizer tool (GCMV)
- Trigger a heap dump and analyze it by using the Memory Analyzer tool
- Examine the diagnostic data artifacts of an application that has a memory leak

Introduction

When the Java virtual machine (JVM) tries to allocate an object and it fails, it runs garbage collection to free up heap space that is occupied by objects that are no longer being used. If the object still cannot be allocated after garbage collection, the JVM throws java.lang.OutOfMemoryError.

In this exercise, you explore some of the tools that are available to monitor a running JVM (Tivoli Performance Viewer). You also use tools to monitor and analyze garbage collection (verbose GC, GCMV), and to analyze Java heap memory dumps (Memory Analyzer).

Requirements

This exercise requires the BadApp application to be installed. This application is installed in Exercise 4: Troubleshooting hung threads.

In addition to the tools available with WebSphere Application Server, you use tools that are available through the IBM Support Assistant. Specifically, you use Garbage Collection and Memory Visualizer tool (GCMV), and the Memory Analyzer tool.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

6.1. Explore and configure tools for gathering JVM heap diagnostic data

In this part of the exercise, you learn how to monitor the heap usage of a server and how to use different tools to generate heap memory dumps. You also learn how to enable lightweight memory leak detection for an application server.

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Start all servers and enable verbose garbage collection for server1

__ 1. If not already started, start the deployment manager and the node agent.

- __ a. From a Terminal window, start the deployment manager by changing to the directory: `<profiles_root>/Dmgr/bin`
- __ b. Enter the command: `./startManager.sh`
- __ c. Start the node agent by changing to the directory: `<profiles_root>/profile1/bin`
- __ d. Enter the command: `./startnode.sh`

__ 2. Start the administrative console.

- __ a. Start a web browser and enter the web address:

  `http://washost:9060/ibm/console`
- __ b. Log on to the administrative console with user ID `wasadmin` and password `web1sphere`
- __ c. If **server1** is started from a previous exercise, stop it now.

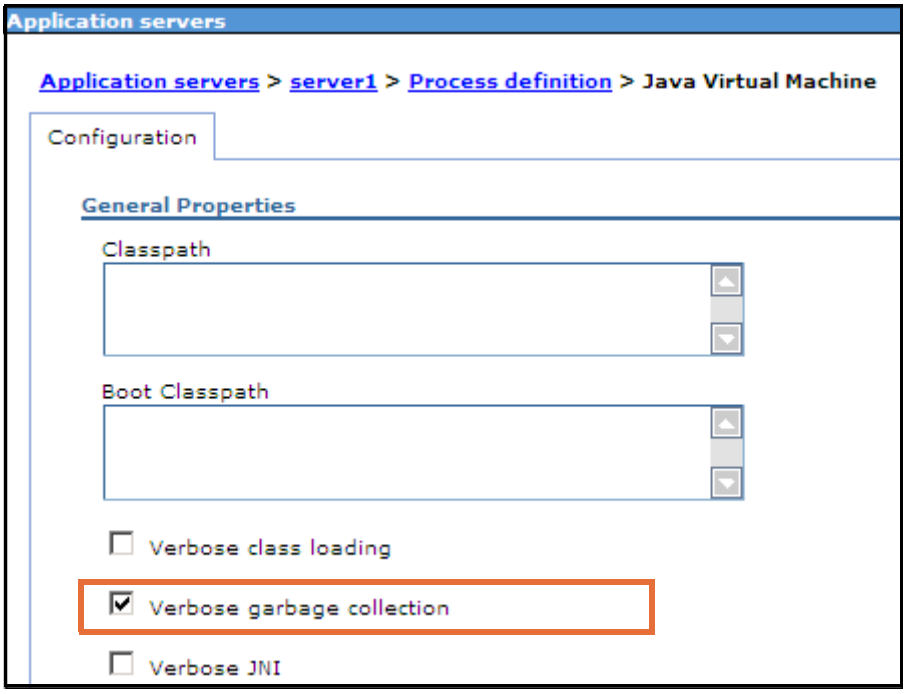
__ 3. Enable verbose GC for server1.

- __ a. Select **Servers > Server Types > WebSphere application server > server1**.
- __ b. On the Configuration tab for server1, scroll down to **Server Infrastructure** and select **Java and Process Management**.
- __ c. Select **Process Definition > Java Virtual Machine**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Check the box for **Verbose garbage collection** and click **OK**.



__ e. Save the changes to the master configuration and wait for node synchronization to complete.

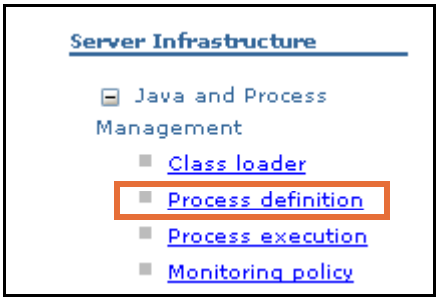
Information

Selecting the box for Verbose garbage collection on the configuration effectively modifies the JVM command line of the server. Specifically, the `verbose:gc` argument is added. Therefore, the server needs to be restarted for this change to take effect. If you need to enable verbose garbage collection for a running instance of a server, you can do so on the Runtime tab.

__ 4. Define the IBM_HEAPDUMPDIR environment variable for server1.

__ a. Select **Servers > Server Types > WebSphere application servers > server1**.

__ b. Under Server Infrastructure, expand **Java Process and Management** and click the **Process definition** link.

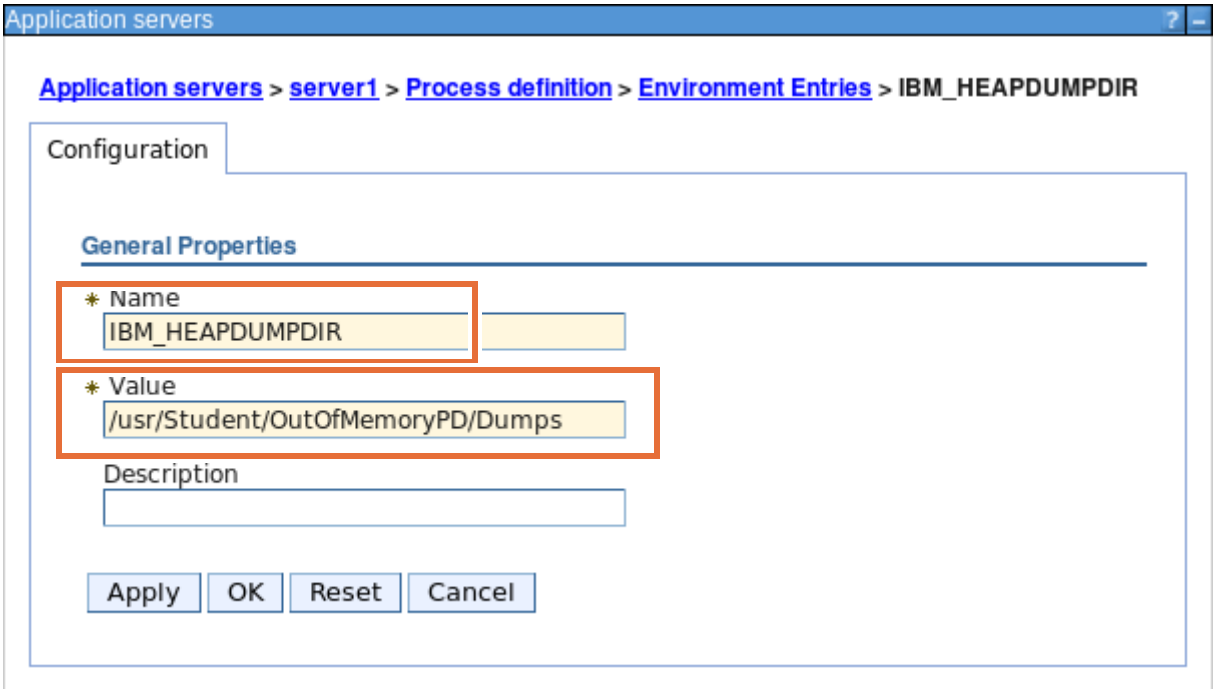


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Under Additional Properties, select **Environment Entries**.

__ a. Click **New**.

__ b. Enter `IBM_HEAPDUMPDIR` for the name, and `/usr/Student/OutOfMemoryPD/Dumps` for the value.



__ c. Click **OK**.

__ d. Click **Save** and wait for the nodes to synchronize. Click **OK**.

__ e. Start **server1** from the administrative console.

Information

The environment variable `IBM_HEAPDUMPDIR` is deprecated in version 8, though it still works. In your environment, you can replace it with the `-Xdump` JVM generic argument.

For example, the following `-Xdump` argument has the same effect as the `IBM_HEAPDUMPDIR` environment variable that you defined:

```
-Xdump:heap:events=user,range=1..1,file=/usr/Student/OutOfMemory/Dumps/heapdump.%Y
%m%d.%H%M%S.%pid.%seq.phd
```

Section 2: Use Tivoli Performance Viewer to monitor the JVM of server1

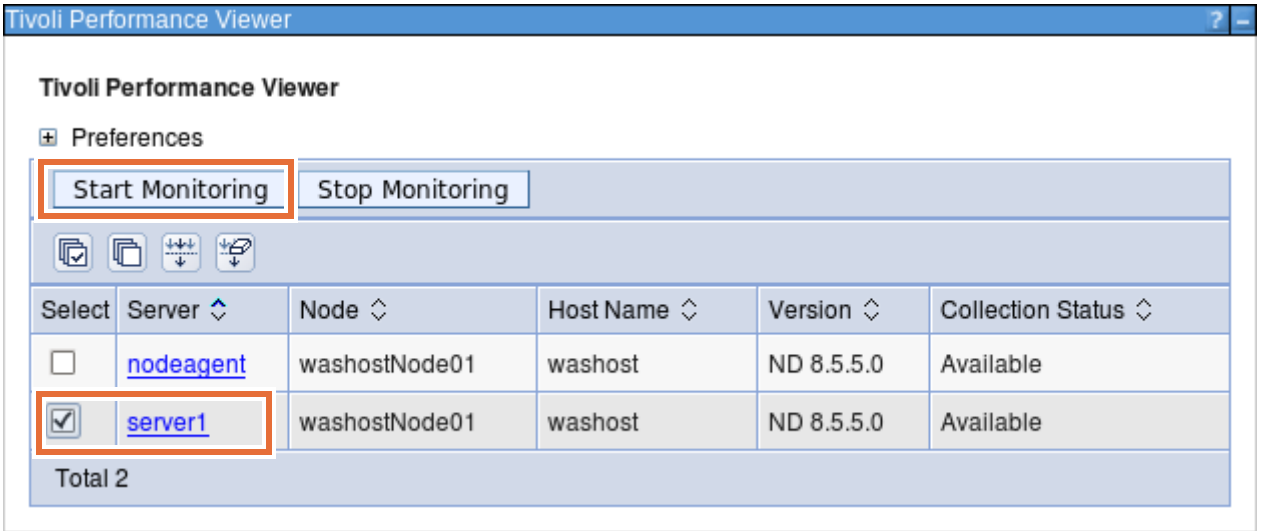
__ 1. Start monitoring server1.

__ a. From the administrative console, select **Monitoring and Tuning > Performance Viewer > Current activity**.

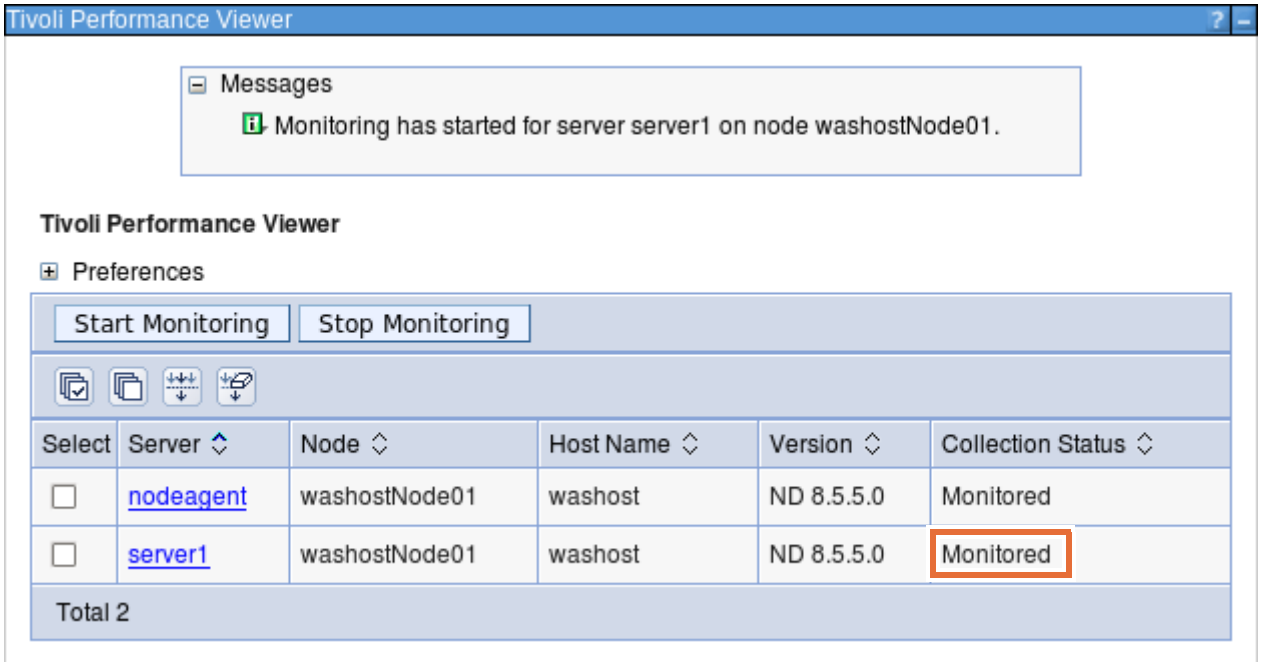
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Check the box for **server1** and click **Start Monitoring**.



__ c. Verify that you see a message that states that "`Monitoring has started for server1...`", and that the Collection Status is changed to **Monitored**.



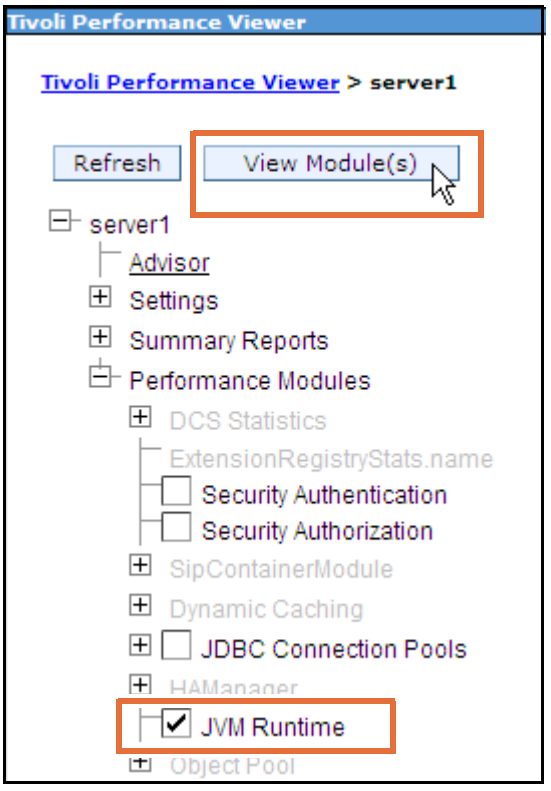
__ 2. Set the Performance Monitor to show the JVM statistics.

__ a. Click the link for **server1** in the window that is shown in the screen capture. This action starts the Tivoli Performance Viewer for server1.

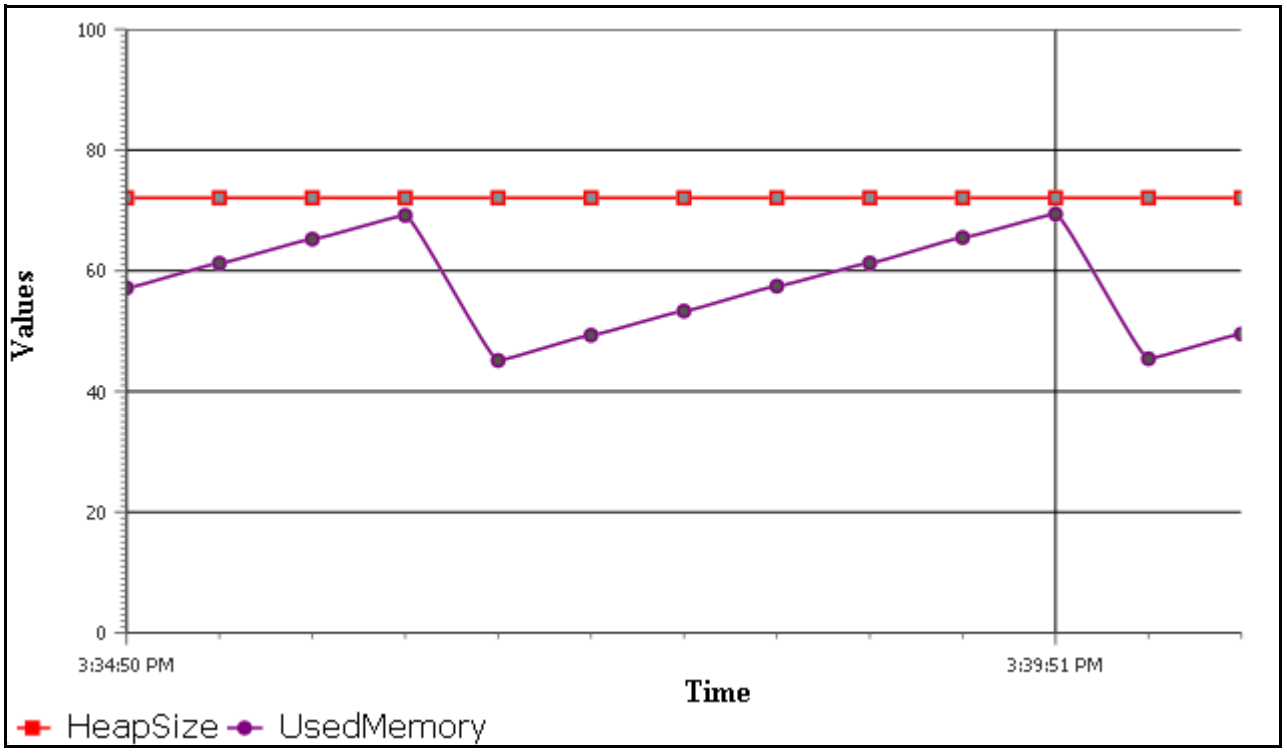
__ b. Click the (**+**) for **Performance Modules** to expand the list.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Check the box for **JVM Runtime** and click **View Modules** near the top of the Tivoli Performance Viewer window.



__ d. Click the **Show Legend** button and clear the check box for **Free Memory**.



__ e. After the monitor runs for several minutes, you see a graph similar to the one shown in the screen capture. This graph depicts a “healthy” JVM. Notice the peaks and valleys (a

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

sawtooth pattern) for the used memory. As memory usage grows over time, garbage collection runs, and memory is freed.

__ f. Keep the Tivoli Performance Viewer running until the end of the next section.

Section 3: Trigger a heap dump for server1 and use the Heap Analyzer tool to examine it

__ 1. Use a wsadmin command to generate a heap dump of server1.

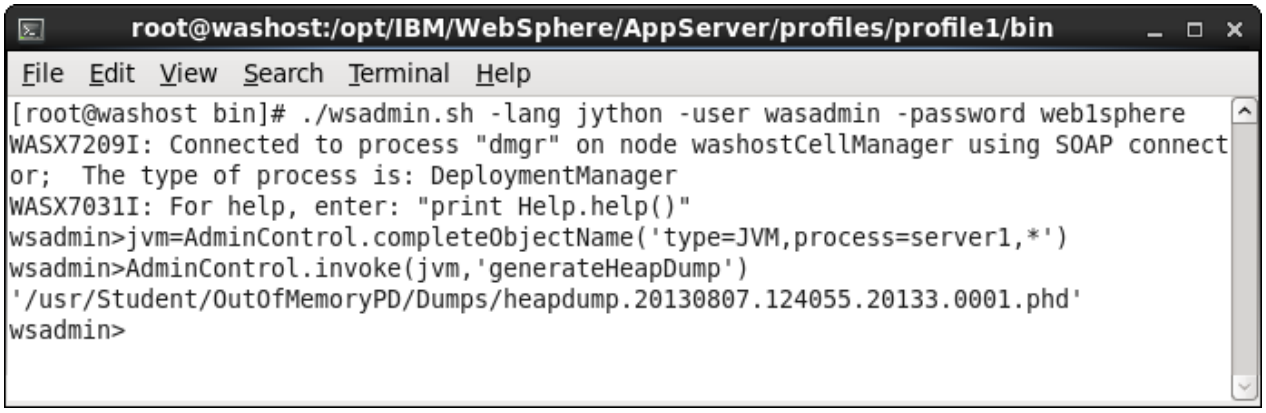
__ a. From a command prompt change directory to: *<profiles_root>*/profile1/bin

__ b. Enter the command:

```
./wsadmin.sh -lang jython -user wasadmin -password web1sphere
```

__ c. At the **wsadmin** prompt, type the following lines (if you prefer, you can copy and paste these lines from the following file /usr/Student/OutOfMemoryPD/wsadmin_cmds.txt):

```
jvm=AdminControl.completeObjectName('type=JVM,process=server1,*')
AdminControl.invoke(jvm,'generateHeapDump')
```



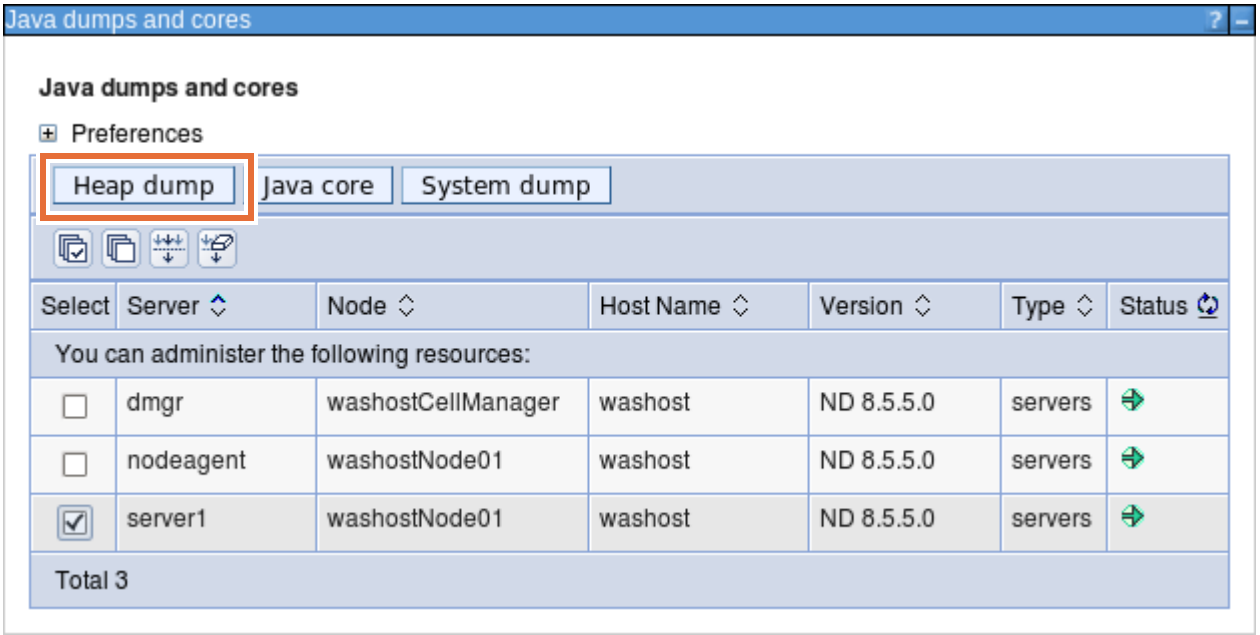
__ d. Wait a few seconds for the dump to complete. Notice that the heap dump file is written to the location that you specified for the IBM_HEAPDUMPDIR environment variable.

__ 2. Use the Heap dump button to generate another heap dump.

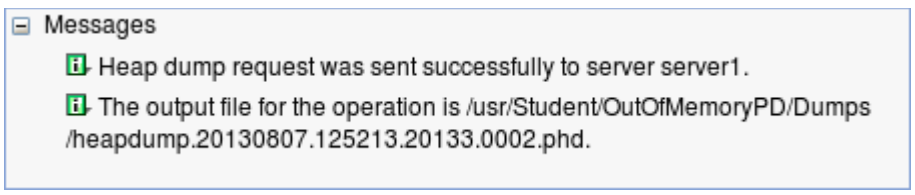
__ a. From the administrative console, click **Troubleshooting > Java dumps and cores.**

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Select **server1** and click **Heap dump**.



__ c. After a few moments, you see the following message in the console.



__ d. By default, heap dumps are in the binary portable heap dump (phd) format. You need a tool such as Heap Analyzer or Memory Analyzer to open them and view their contents. In the next part of this exercise, you use the Memory Analyzer to analyze heap dumps.

__ 3. View the JVM in the Tivoli Performance Viewer.

__ a. From the administrative console click **Monitoring and Tuning > Performance Viewer > Current Activity > server1**.

__ b. You see the sawtooth pattern as shown in the earlier screen capture and maybe some fluctuations at the times when you triggered the heap dumps.

__ 4. Stop the performance monitor.

__ a. From the administrative console, select **Monitoring and Tuning > Performance Viewer > Current Activity**.

__ b. Check the boxes for **server1** and **nodeagent**, and click **Stop Monitoring**.

__ c. Verify that you see messages similar to the following for the node agent and server1:

```
Monitoring has stopped for server nodeagent on node washostNode01.
Monitoring has stopped for server server1 on node washostNode01.
```


© Copyright IBM Corp. 2013

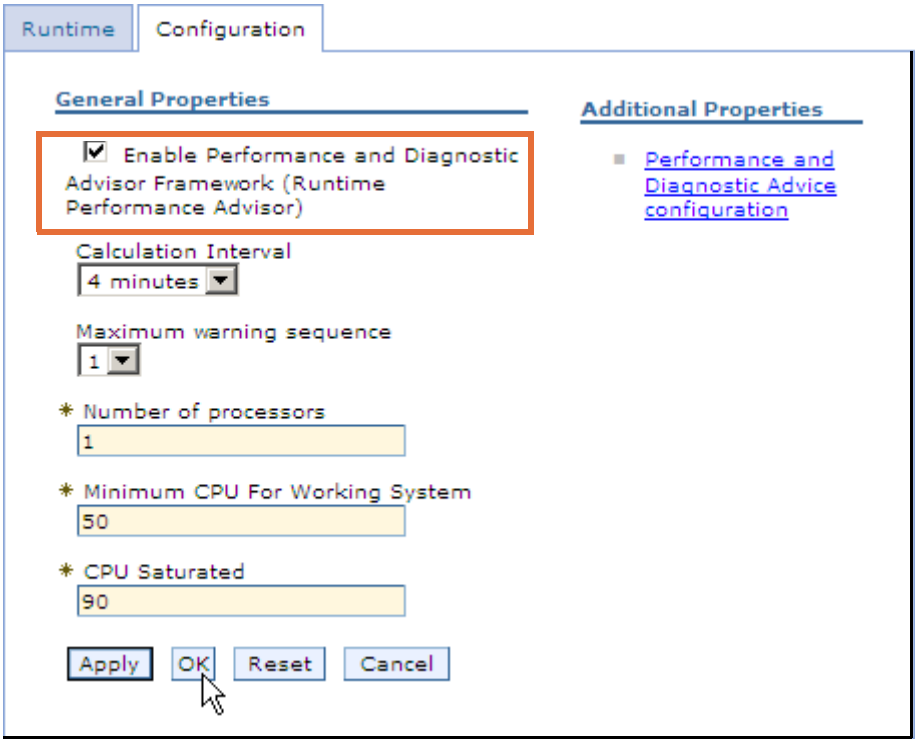
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 5. Configure WebSphere lightweight memory leak detection mechanism.

__ a. From the administrative console, click **Servers > Server Types > WebSphere application servers > server1**.

__ b. On the configuration tab of server1, scroll down to the **Performance** section and click the link for **Performance and Diagnostic Advisor Configuration**.

__ c. On the Configuration tab, select the check box for **Enable Performance and Diagnostic Advisor Framework**.



__ d. Click **OK**.

__ e. Click **OK** on the confirmation message.

__ f. Click **Save** to save changes to the master configuration.

__ g. Wait for the nodes to synchronize and click **OK**.

__ 6. Verify that leak detection is started

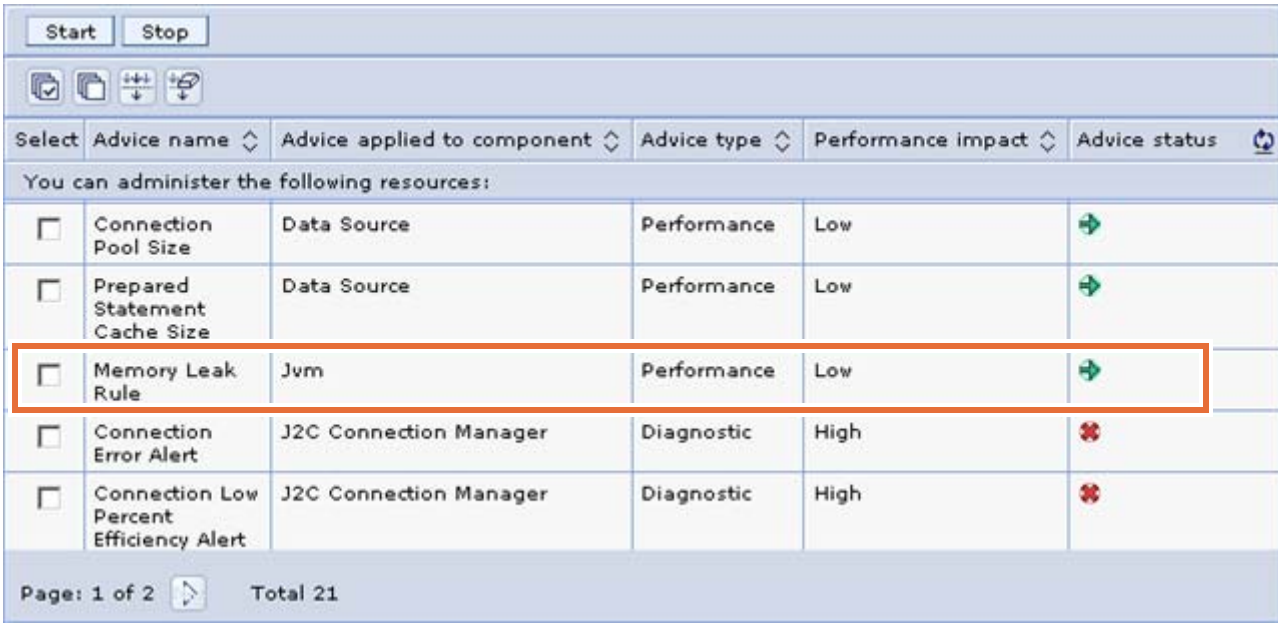
__ a. Click **Servers > Server Types > WebSphere application servers > server1**.

__ b. On the configuration tab of server1, scroll down to the **Performance** section and click the link for **Performance and Diagnostic Advisor Configuration**.

__ c. Under Additional Properties, click the link **Performance and Diagnostic Advice configuration**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Scroll down to **Memory Leak Rule**. Verify that its Advice status is started.



Information

The Performance and Diagnostic Advisor analyzes PMI data and receives notifications about performance and diagnostic information from components. Use this page to specify settings for the Performance and Diagnostic Advisor. Performance issues can be related to memory leaks in the system. Use the Memory Analyzer tool, a separate memory leak analysis utility, for detecting memory leaks.

WebSphere implements a lightweight memory leak detection mechanism. This mechanism is designed to provide early detection of memory problems in test and production environments. This framework is not designed to provide analysis of the source of the problem, but rather to provide notification and help for generating the information that is required to use analysis tools. The mechanism detects memory leaks in the Java heap only and does not detect native memory leaks.

The lightweight memory leak detection in WebSphere Application Server does not require any additional agents. The detection relies on algorithms that are based on information that is available from the Performance Monitoring Infrastructure service and has minimal effect on performance.

It is suggested that you run the Performance and Diagnostic Advisor in a production simulation and test environment. Performance advice is most applicable during peak load, when the processor usage is high.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

6.2. Troubleshoot an application that has a Java out-of-memory error

In this part of the exercise, you run the BadApp application, which has a built-in coding problem that results in heap exhaustion and an OutOfMemoryError being thrown. You use various tools and techniques to determine what the problem is for which you are seeing symptoms. You use the Memory Analyzer to determine what data structures in the application might be causing the problem.

Section 1: Run the BadApp application and generate the OutOfMemoryError exception

__ 1. Clean up the native log files and the heap dump files that you created in the last section.

- __ a. Stop **server1**.
- __ b. Navigate to `<profiles_root>/profile1/logs/server1` and delete the `native_stderr.log` and `native_stdout.log` files.
- __ c. Navigate to `/usr/Student/OutOfMemoryPD/Dumps` and delete all the files in the directory.
- __ d. Start **server1**.

__ 2. Trigger a heap dump of server1.

- __ a. From the administrative console, click **Troubleshooting > Java dumps and cores**.
- __ b. Select **server1** and click **Heap dump**.
- __ c. Navigate to `/usr/Student/OutOfMemoryPD/Dumps` and change the heap dump file name to: `heapdump_beforeOOM.phd`

__ 3. Enable Tivoli Performance Viewer monitoring for the JVM of server1 as you did in the first part of this exercise in **Section: 2 Use Tivoli Performance Viewer to monitor the JVM of server1**. Resize the browser so that you can view it along with other browsers and windows.

__ 4. Use the command line Log Viewer to monitor the log for server1.

- __ a. Open a command prompt window and change directory to: `<profiles_root>/profile1/bin`
- __ b. Enter the command: `./logViewer.sh -monitor`

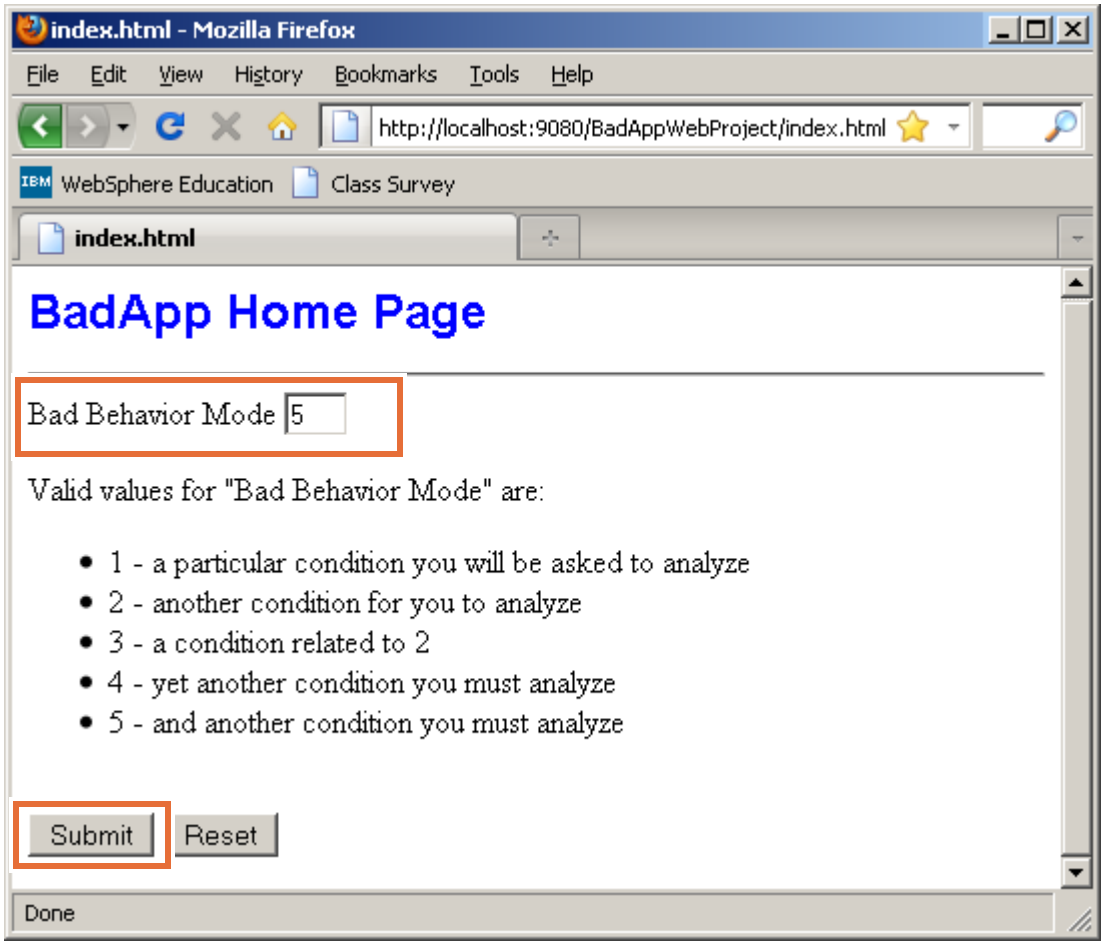
__ 5. Run the BadApp enterprise application.

- __ a. Start a browser and enter the web address:

  `http://localhost:9080/BadAppWebProject/index.html`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter 5 for the **Bad Behavior Mode**, and click **Submit**.



__ 6. The browser hangs for about 6 minutes until the application displays an error page. During this time, you can perform a number of actions to monitor the behavior of the application.

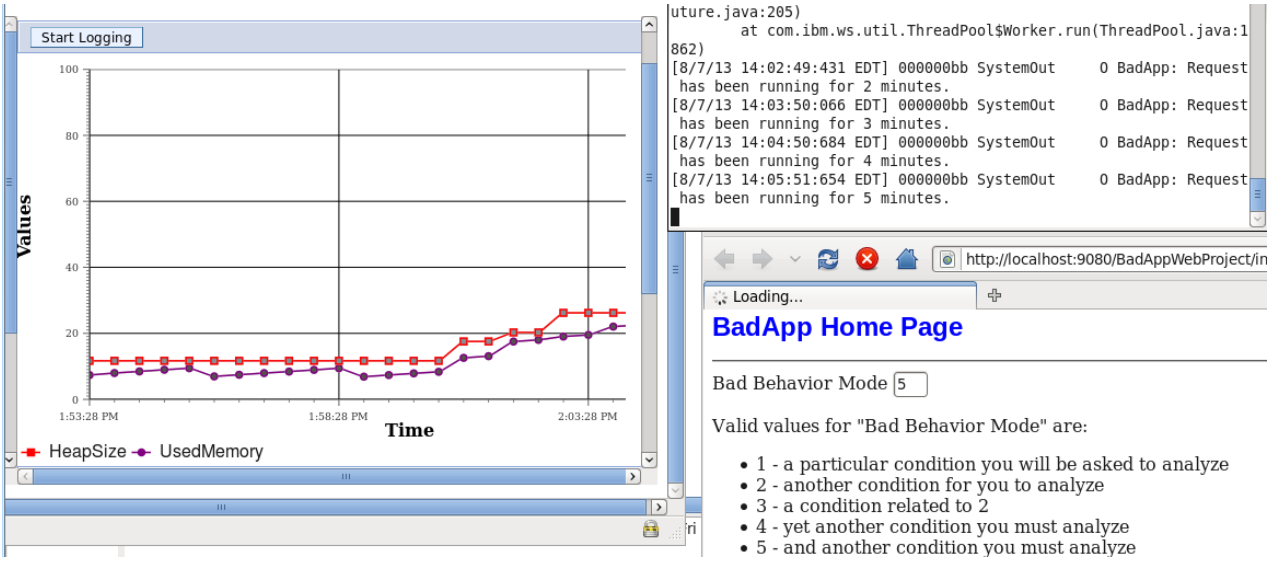
__ a. Use Tivoli Performance Viewer to monitor the server1 JVM **HeapSize** and **UsedMemory**. Clear other metrics such as FreeMemory.

__ b. You might be required to reset the scales for HeapSize and UsedMemory. They should both be scaled to 0.0001.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

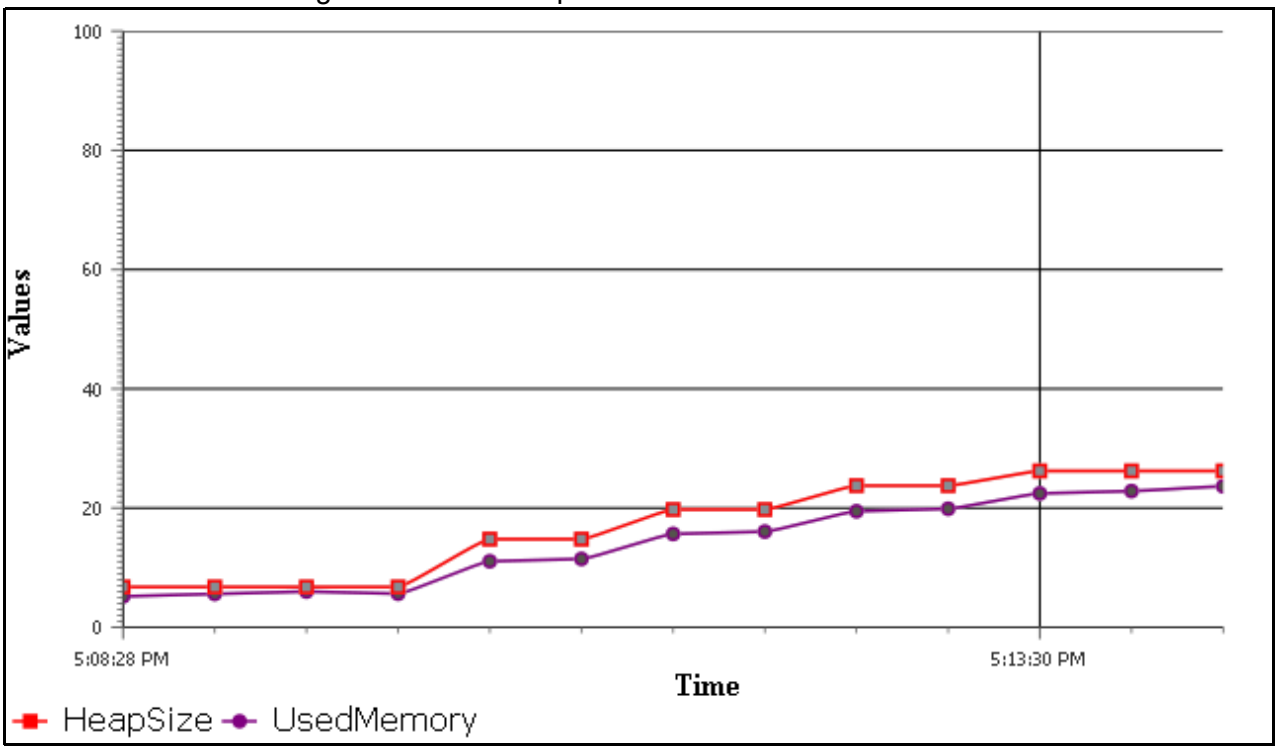
__ c. Use the Terminal window to monitor log messages.



__ d. After several minutes, you see the BadApp return page.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Examine the Tivoli Performance Viewer. You see that the heap size and used memory are increasing to maximum heap size.



__ 7. Review the log messages in the command window.

__ a. Scroll to the beginning of the log and notice the **SystemOut** logger messages.

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
************* End Display Current Environment *************
[8/8/13 10:22:02:210 EDT] 0000009a ServletWrappe I com.ibm.ws.webcontainer.servl
et.ServletWrapper init SRVE0242I: [BadAppEARProject] [/BadAppWebProject] [com.ib
m.issf.atjolin.badapp.BadAppServlet]: Initialization successful.
[8/8/13 10:22:02:216 EDT] 0000009a SystemOut     O   BadApp: Parameters passed f
rom the user:
[8/8/13 10:22:02:216 EDT] 0000009a SystemOut     O   BadApp: Parameter badBehavi
orMode = 5
[8/8/13 10:23:11:942 EDT] 00000034 ThreadMonitor W   WSVR0605W: Thread "WebConta
iner : 0" (0000009a) has been active for 69,763 milliseconds and may be hung.  T
here is/are 1 thread(s) in total in the server that may be hung.
        at java.lang.Thread.sleep(Native Method)
        at java.lang.Thread.sleep(Thread.java:897)
        at com.ibm.issf.atjolin.badapp.BadAppServlet.docMethod(BadAppServlet.jav
a:417)
        at com.ibm.issf.atjolin.badapp.BadAppServlet.doPost(BadAppServlet.java:2
80)
        at javax.servlet.http.HttpServlet.service(HttpServlet.java:595)
        at javax.servlet.http.HttpServlet.service(HttpServlet.java:668)
        at com.ibm.ws.webcontainer.servlet.ServletWrapper.service(ServletWrapper
```

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

These messages are written to the stdout stream by the BadApp application. They show that Mode 5 was selected, and the request was running for over a minute.

Since the thread monitor is configured with a 60-second threshold, it logs a message that says that the “WebContainer:0 thread may be hung”, and dumps a stack trace.

__ b. Scroll down to last SystemOut O BadApp message that says:

`hit an OutOfMemoryError (maybe as expected)`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
        at com.ibm.ws.util.ThreadPool$Worker.run(ThreadPool.java:1862)
[8/8/13 10:23:13:077 EDT] 0000009a SystemOut     O   BadApp: Request has been ru
nning for 1 minutes.
[8/8/13 10:24:13:310 EDT] 00000090 authz         I   CWWIM2000I Initialization o
f the authorization component completed successfully.
[8/8/13 10:24:13:878 EDT] 0000009a SystemOut     O   BadApp: Request has been ru
nning for 2 minutes.
[8/8/13 10:25:14:623 EDT] 0000009a SystemOut     O   BadApp: Request has been ru
nning for 3 minutes.
[8/8/13 10:26:15:224 EDT] 0000009a SystemOut     O   BadApp: Request has been ru
nning for 4 minutes.
[8/8/13 10:27:17:045 EDT] 0000009a SystemOut     O   BadApp: Request has been ru
nning for 5 minutes.
[8/8/13 10:27:47:964 EDT] 0000009a SystemOut     O   BadApp: hit an OutOfMemoryE
rror (maybe as expected)
[8/8/13 10:27:47:967 EDT] 0000009a SystemErr     R   java.lang.OutOfMemoryError:
 Java heap spacenull
[8/8/13 10:27:47:968 EDT] 0000009a SystemErr     R      at com.ibm.issf.atjolin.
badapp.BadAppServlet$F.<init>(BadAppServlet.java:192)null
[8/8/13 10:27:47:968 EDT] 0000009a SystemErr     R      at com.ibm.issf.atjolin.
badapp.BadAppServlet.docMethod(BadAppServlet.java:406)null
[8/8/13 10:27:47:968 EDT] 0000009a SystemErr     R      at com.ibm.issf.atjolin.
badapp.BadAppServlet.doPost(BadAppServlet.java:280)null
```

You might see a “CPU Starvation detected” message from the HA manager or AlarmThread messages. But also notice the `SystemErr R java.lang.OutOfMemoryError: Java heap spacenull`.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Scroll down to the next thread monitor message that says:

`Thread "WebContainer : 0" (0000009a) was previously reported to be hung has completed.`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
et.ServletWrapper init SRVE0242I: [BadAppEARProject] [/BadAppWebProject] [/BadAp
pJSP.jsp]: Initialization successful.
[8/8/13 10:27:48:350 EDT] 0000009a ThreadMonitor W   WSVR0606W: Thread "WebConta
iner : 0" (0000009a) was previously reported to be hung but has completed.  It w
as active for approximately 346,170 milliseconds.  There is/are 0 thread(s) in t
otal in the server that still may be hung.
[8/8/13 10:28:59:005 EDT] 0000008f TraceResponse W   TUNE9001W:  Heap utilizatio
n patterns indicate that you may have a memory leak
Additional explanatory data follows.
Data values for free memory between 8/8/13 10:27 AM and 8/8/13 10:28 AM were con
sistently below minimum required percentage.
```

Notice that the Thread monitor also reports how long the thread was active, in this example, approximately 346,170 ms.

Finally, notice the `TUNE9001W` message. This message is from the lightweight memory leak detection mechanism that you configured by using the Performance Advisor.

__ 8. Check for a heap dump that the OutOfMemoryError triggered.

__ a. Navigate to: `/usr/Student/OutOfMemoryPD/Dumps`

You see a second heap dump file.

__ b. Rename this second heap dump file: `heapdump_afterOOM.phd`

__ c. You might need to do a comparative analysis of these two heap dumps later in this exercise.

__ 9. Examine the verbose GC messages for server1.

__ a. Navigate to: `<profiles_root>/profile1/logs/server1`

__ b. Open the `native_stderr.log` file with gedit.

__ c. Search for the message code `JVMDUMP039I` and you see messages such as

```
JVMDUMP039I Processing dump event "systhrow", detail "java/lang/OutOfMemoryError"
at 2014/02/04 12:48:01 - please wait.
JVMDUMP032I JVM requested Java dump using
'/opt/IBM/WebSphere/AppServer/profiles/profile1/javacore.20140204.124801.3933.00
09.txt' in response to an event
JVMDUMP010I Java dump written to
/opt/IBM/WebSphere/AppServer/profiles/profile1/javacore.20140204.124801.3933.000
9.txt
```

__ d. Copy the System dump file to: `/usr/Student/OutOfMemoryPD/Dumps`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Look in `native_stderr.log` for the allocation failure that occurred just before the OutOfMemoryError message. From this log entry, you see important data.

```
<gc-end id="2666" type="global" contextid="2659" durationms="485.229"
timestamp="2014-02-04T12:46:57.269">
  <mem-info id="2667" free="514784" total="258146304" percent="0">
    <mem type="nursery" free="361600" total="56885248" percent="0" />
    <mem type="tenure" free="153184" total="201261056" percent="0">
      <mem type="soa" free="153184" total="201261056" percent="0" />
      <mem type="loa" free="0" total="0" percent="0" />
    </mem>
    <remembered-set count="2669" />
  </mem-info>
</gc-end>
<cycle-end id="2668" type="global" contextid="2659"
timestamp="2014-02-04T12:46:57.269" />
<allocation-unsatisfied id="2669" threadId="0000000014712900"
bytesRequested="1024008" />
<af-end id="2670" timestamp="2014-02-04T12:46:57.269" />
<exclusive-end id="2671" timestamp="2014-02-04T12:46:57.269"
durationms="1328.969" />
```

First, notice that the value of **bytesRequested** is 1024008. If you then look at how much memory is available after the garbage collection, you can see why the OutOfMemory exception was thrown and the result of the allocation is **unsatisfied**.

__ f. Since this exercise is a controlled test situation, it is suggested that you stop server1 from the administrative console. Because of the memory exhaustion, the server is likely to be unresponsive to an ordinary Stop command. Use the **Terminate** button.

Section 2: Use tools to analyze OutOfMemoryError exception artifacts

__ 1. Make sure that IBM Support Assistant is started. You started it in the previous sections, but if you stopped it, follow these steps:

__ a. Start the IBM Support Assistant by clicking **Applications** on the toolbar. Select **Start IBM Support Assistant 5**.

__ b. A terminal window pops up. Wait until you see a message that says the IBM Support Assistant is started. Do not close this window until you are finished with the IBM Support Assistant.

__ 2. Launch the **Garbage Collection and Memory Visualizer (GCMV) [Desktop]** from the **IBM Support Assistant**.

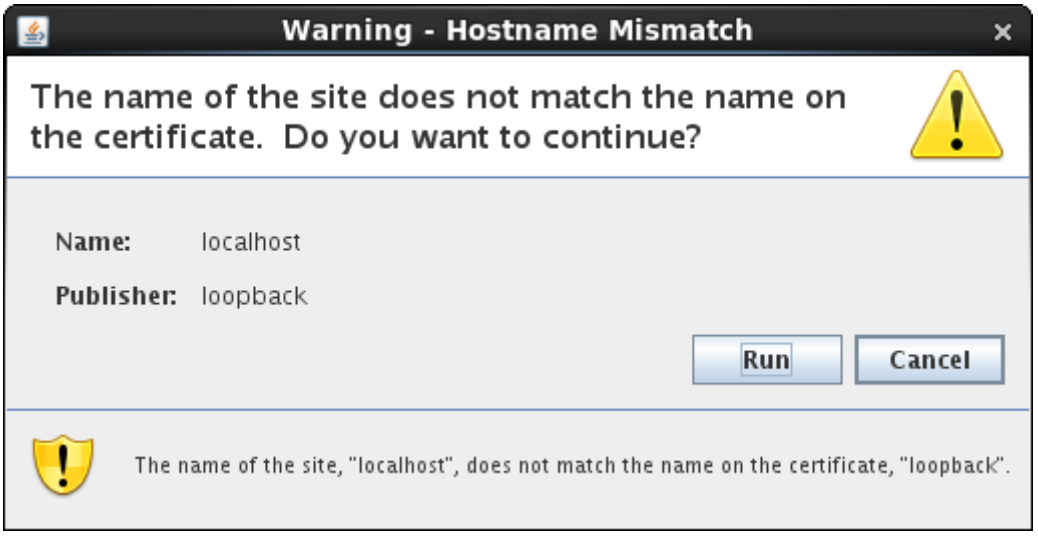
__ a. Open a web browser, and enter the web address:

__ a. `http://washost:10911/isa5`

__ b. You can launch the **GCMV [Desktop]** from the **Tools** tab of the IBM support Assistant.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Click **Run** if prompted about a hostname mismatch.



__ d. Wait for Java Web Start to download the application and for the GCMV tool to start.

__ 3. Import the GC logs.

__ a. In the GCMV tool, click **File > Load file**.

__ b. Navigate to the `<profiles_root>/profile1/logs/server1` directory and select `nativ_stderr.log`.

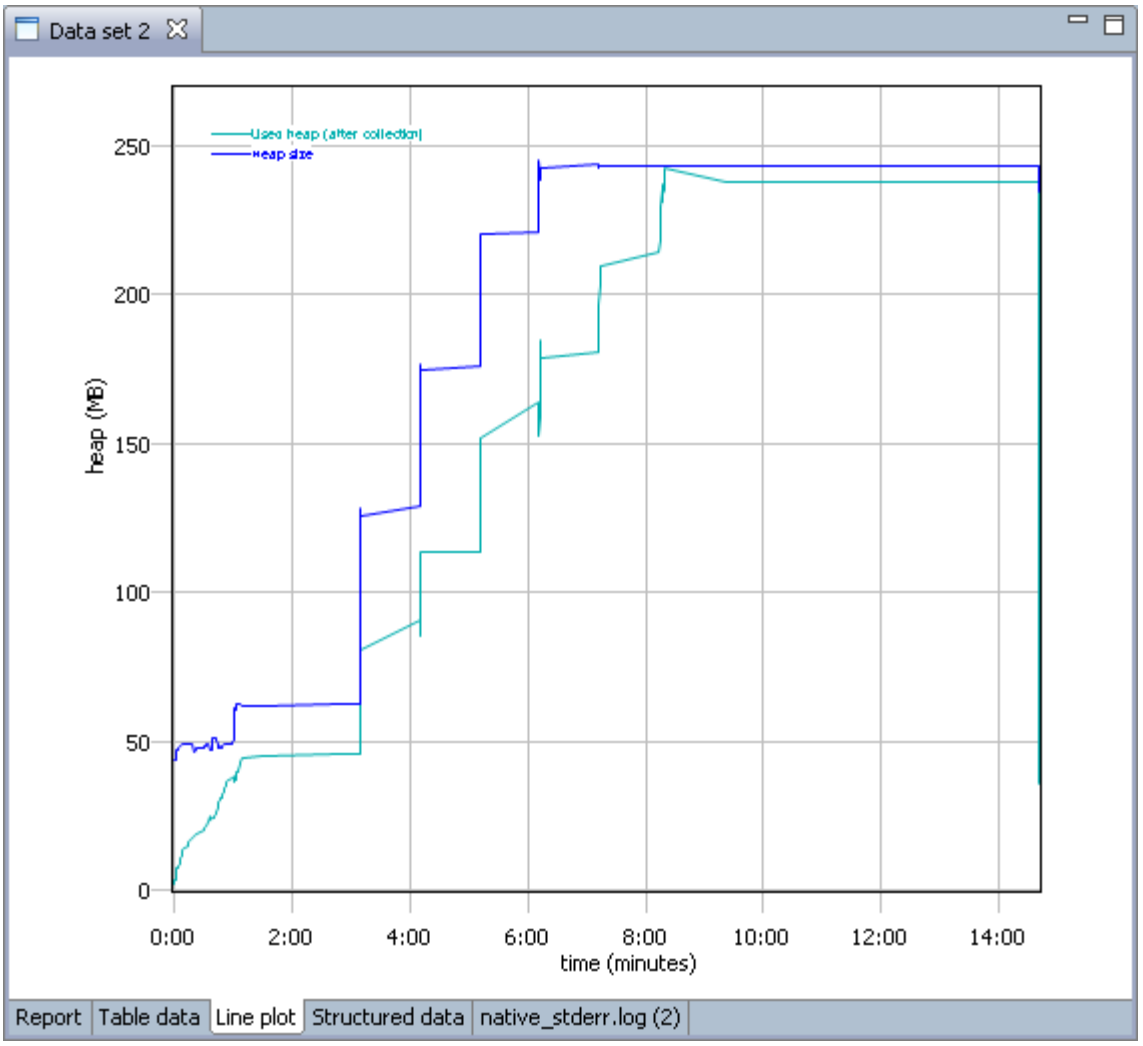
__ c. Click **OK**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 4. Explore the analysis of the verbose GC data.

__ a. Select the **Line plot** tab. You see a graph similar to the following plot.



Note

The top line is the heap size. The bottom line is the used heap (after GC collection).

__ b. Notice that the heap size expanded to approximately 250 MB. What is the maximum heap setting for server1? What happened to the free heap?

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select the **Report** tab and examine the details of the report.

__ d. View the Tuning recommendation.

Tuning recommendation

Your application appears to be leaking memory. This is indicated by the used heap increasing at a greater rate than the application workload (measured by the amount of data freed). To investigate further see Guided debugging for Java

The Java Heap has been exhausted, leading to an out of memory error. You should consider increasing the Java Heap size using -Xmx if space allows. You can analyse the usage of the Java Heap for a memory leak by using the ISA Tool Add-on, IBM Monitoring and Diagnostic Tools for Java - Memory Analyzer.

The application seems to be using some quite large objects. The largest request which triggered an allocation failure (and was recorded in the verbose gc log) was for 5120008 bytes.

__ e. Your recommendations might be slightly different from the ones that are shown here, but you see two types of messages, problems (red “X”) and warnings (Yellow “!”). The problem-type messages indicate that the application is leaking memory and that the Java heap is exhausted. Scroll through the Tuning recommendation section and examine some of the other recommendations.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. Scroll down and view the **Summary**.

Summary

Concurrent collection count	83
Forced collection count	2
GC Mode	gencon
Global collections - Mean garbage collection pause (ms)	303
Global collections - Mean interval between collections (minutes)	0.04
Global collections - Number of collections	361
Global collections - Total amount tenured (MB)	83339
Largest memory request (bytes)	5120008
Number of collections triggered by allocation failure	534
Nursery collections - Mean garbage collection pause (ms)	17.6
Nursery collections - Mean interval between collections (ms)	4338
Nursery collections - Number of collections	258
Nursery collections - Total amount flipped (MB)	1960
Nursery collections - Total amount tenured (MB)	1123
Proportion of time spent in garbage collection pauses (%)	12.2
Proportion of time spent unpaused (%)	87.8
Rate of garbage collection (MB/minutes)	154

Though the information that this summary provides is valuable for monitoring and tuning JVM performance, there is not much here that would help you pinpoint the root cause of an OutOfMemory exception. The “Largest memory request” metric can be helpful. However, you would need to gather more diagnostic data about the Java heap contents and the number and size of the objects that are stored on the heap at the time of the OutOfMemory exception.

One of the tuning recommendations suggests the Memory Analyzer tool to analyze the heap. You use the Memory Analyzer tool in the next step.

__ g. When you are done exploring the analysis, close **GCMV** by clicking **File > Exit**.

__ 5. Use the Memory Analyzer tool to analyze the heap dump that was generated when the OutOfMemory exception was thrown. This tool provides the necessary diagnostic data for

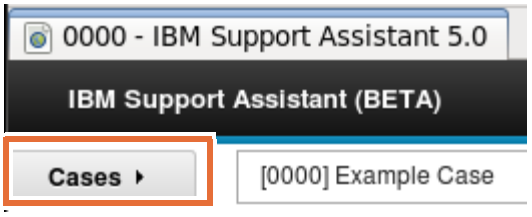
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

pinpointing the root cause of an OutOfMemory exception. Make sure that IBM Support Assistant is started.

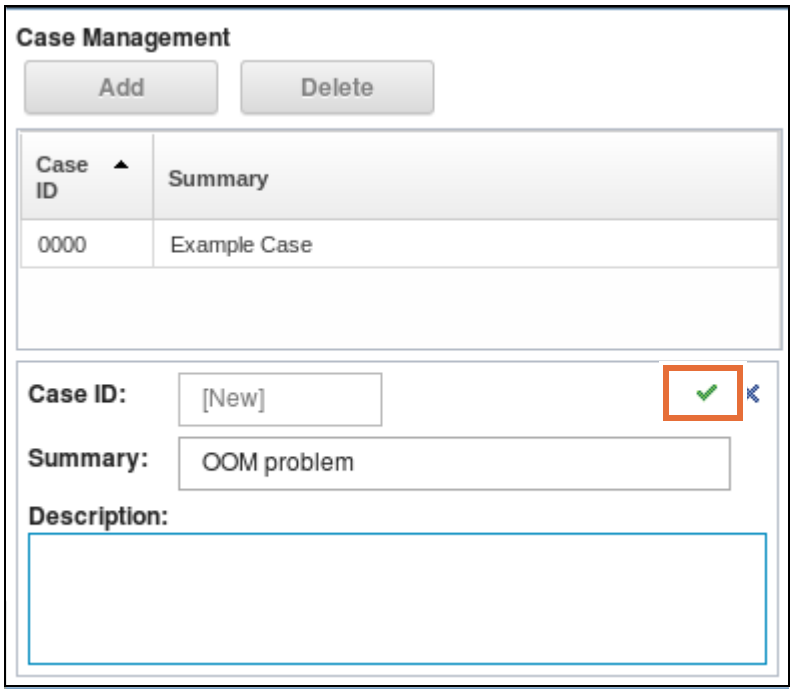
__ a. Open a web browser, and enter the web address: `http://washost:10911/isa5`

__ b. Wait for the IBM Support Assistant home page to show up in the browser.

__ c. Create a case by clicking **Cases** to open the Case Management pane.



__ d. In the Case Management pane, click **Add** and enter `OOM problem` for Summary.



__ e. Click the green check mark to save. The Case ID is assigned automatically.

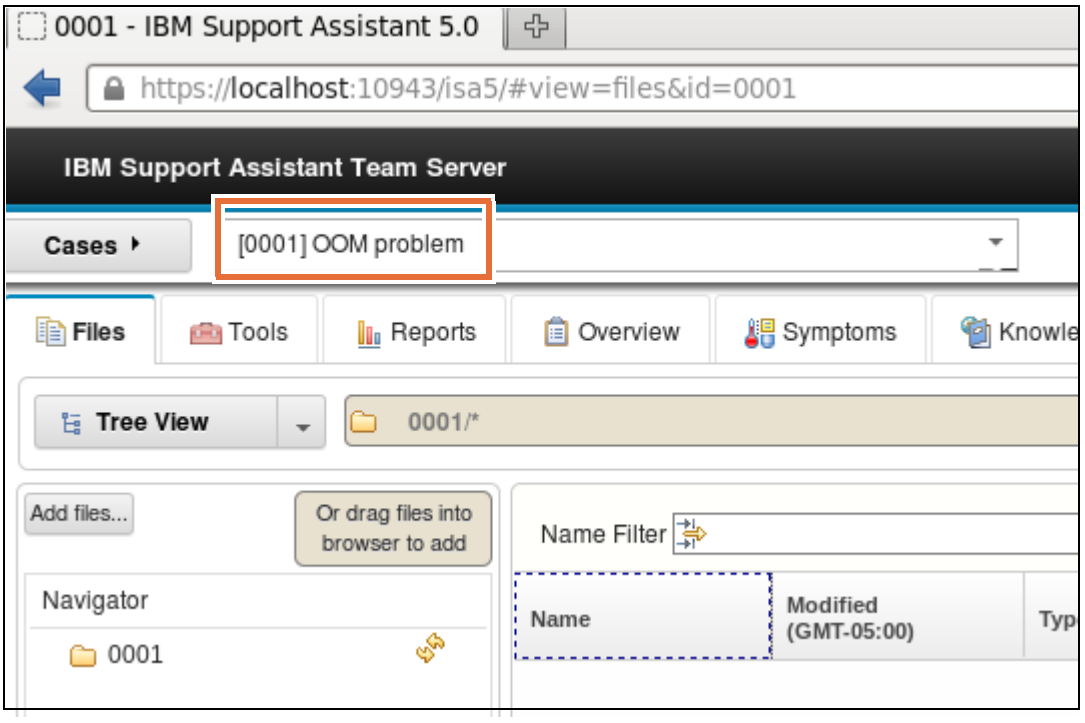
__ f. Click anywhere outside of the **Case Management** pane to make it go away.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 6. Add the heap dump files to the case.

__ a. Make sure that the case name **OOM problem** shows up beside the **Cases** button.



__ b. Click **Add files**.

__ c. Navigate to `/usr/Student/OutOfMemoryPD/Dumps` and select `heapdump_afterOOM.phd`

__ d. Click **Open**.

__ 7. Run the web version of the Memory Analyzer. You use the Memory Analyzer Web Edition to analyze a heap dump in this section and a system core dump in the next section. The Memory Analyzer tools use large amounts of physical memory when they analyze large heap dumps and core dumps. The Memory Analyzer Java process can grow to more than 1 GB during an analysis.The Memory Analyzer [Web] server is started whenever you start IBM Support Assistant.

__ a. Click the **Tools** tab.

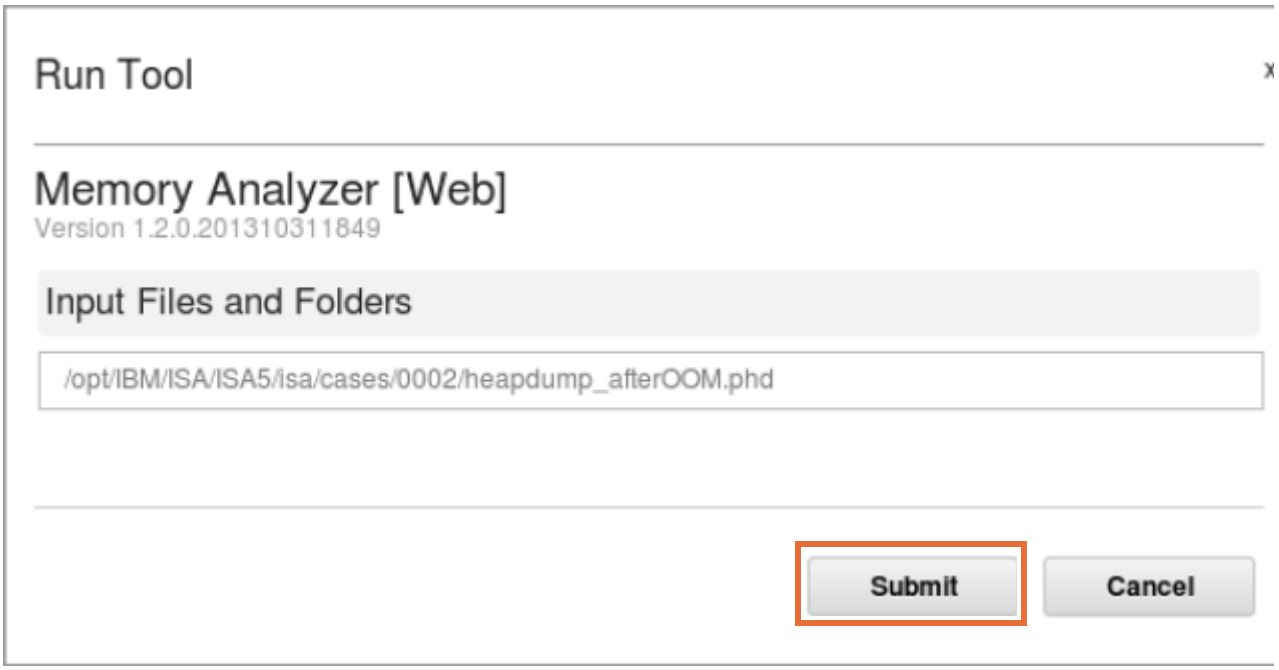
__ b. Select **Memory Analyzer Web Edition** from the list of tools.

__ c. In the right pane for Memory Analyzer, click **Launch**.

__ d. Click **Submit**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. On the Problem Analysis pane, verify the dump file and click **Submit**.



Note

Web browser security for Web Edition tools

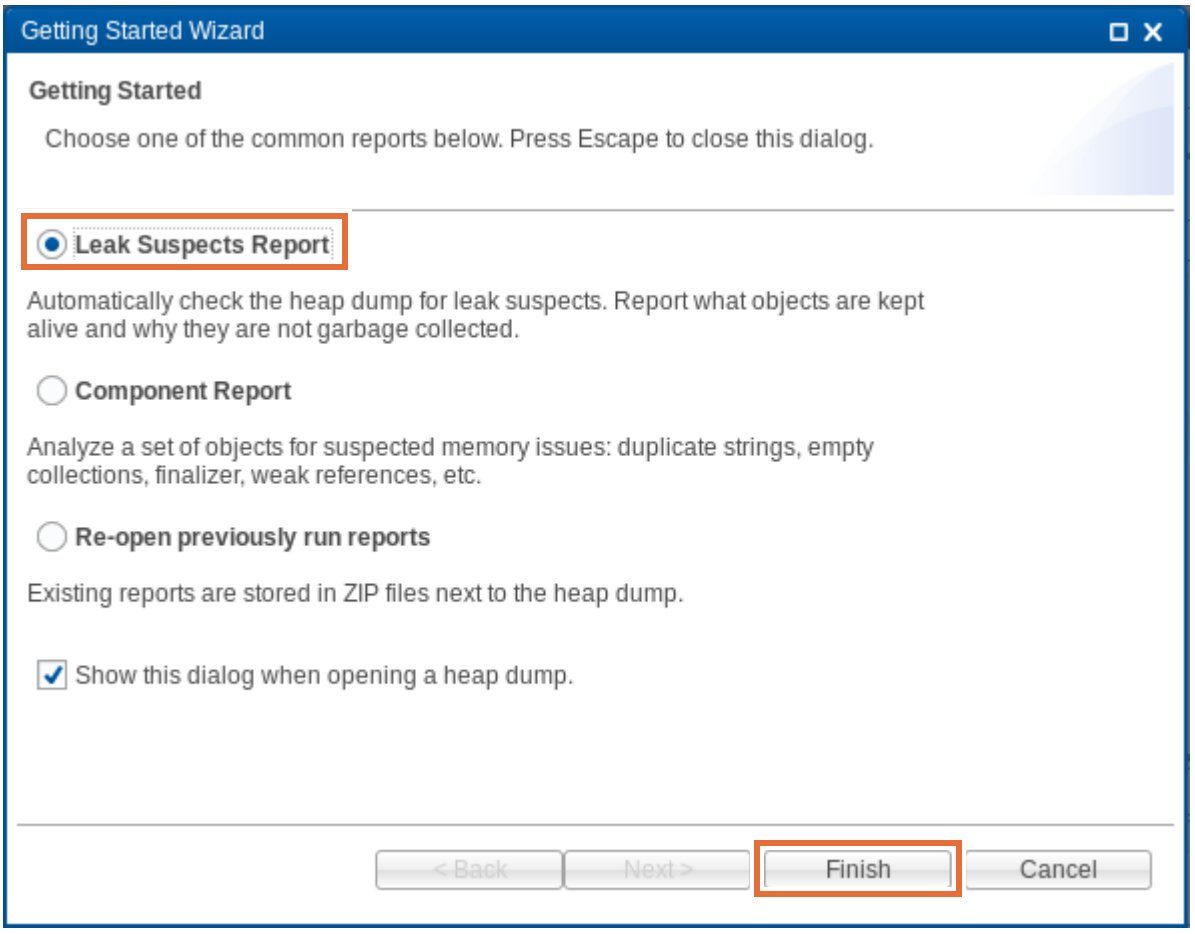
The first time that you run a web type tool, your browser might block the pop-up. If that happens, direct the browser to accept the pop-up. Launch the Memory Analyzer again if necessary.

You might also see a "**This Connection is Untrusted**" warning. If so, click **I understand the Risks** and click **Add Exception**. Then, click **Confirm Security Exception**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Wait a few minutes for the tool to process the heap dump. Then, select **Leak Suspects Report** on the Getting Started wizard.



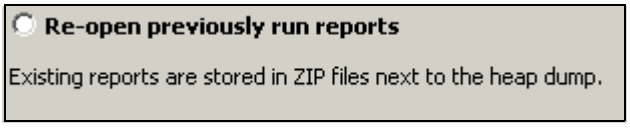
Note

The processing of the heap dump can take more than 5 minutes if the file is large.

__ g. Click **Finish**, and wait for the generated report.

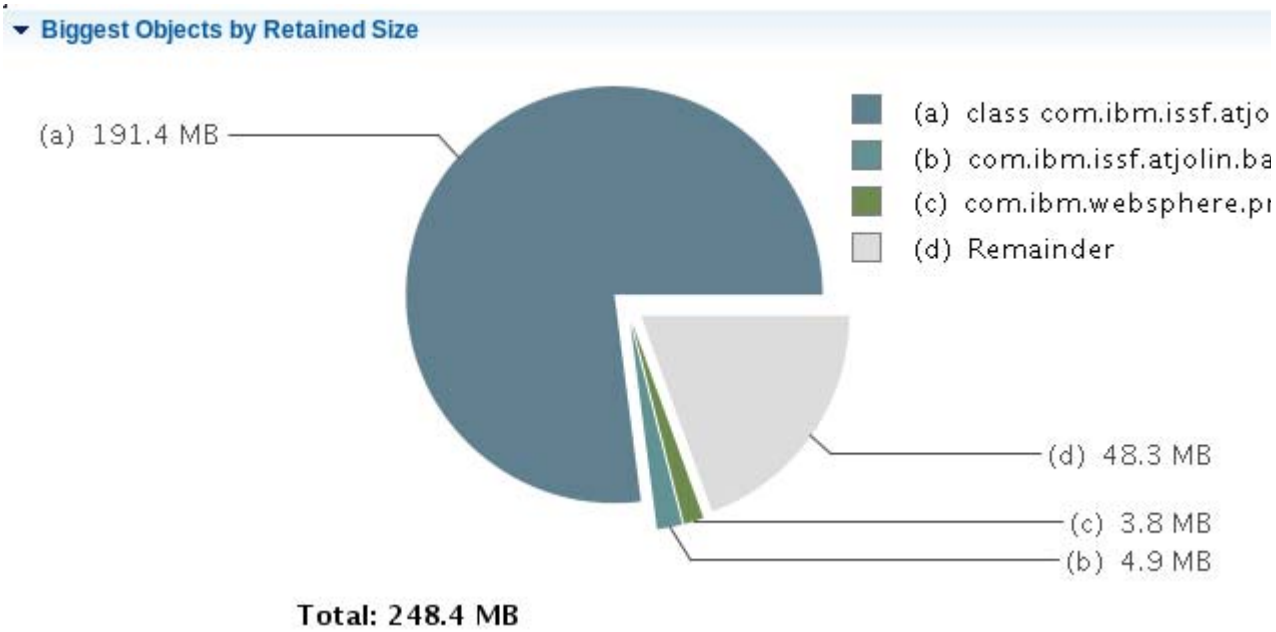
Information

After you perform a heap dump analysis in Memory Analyzer, the analysis is saved and you can view it later by selecting **Reopen Previously run reports** in the Getting Started pane.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ h. Select the **Overview** tab to see the Biggest Objects pie chart.



__ i. Notice that the biggest object by retained size is identified as the BadAppServlet. It uses 185 MB of the total memory of 242 MB.

Information

A note on Java heap terminology:

Shallow heap: The amount of memory that one object consumes. Depending on the operating system architecture, an object needs 32 bits or 64 bits per reference, 4 bytes per “Integer”, 8 bytes per “Long”, and others. Depending on the heap dump format, the size can be adjusted to provide a more realistic consumption of the JVM.

Retained set: One or more objects plus any objects that are referenced, directly or indirectly, only from those original objects. The retained set is the set of objects that garbage collection removes when an object, or multiple objects, are garbage collected.

Retained heap, or retained size: The total heap size of all the objects in the retained set. This value is the amount of memory that is consumed by all the objects that are kept alive by the objects at the root of the retained set. In general terms, the **shallow heap** of an object is the size of the object in the heap. The **retained size** of the same object is the amount of heap memory that is freed when the object is garbage collected.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ j. Next, select the **default_report** tab. Examine the Leak Suspects section to see more details about the BadAppServlet object.

Problem Suspect 1

The class **"com.ibm.issf.atjolin.badapp.BadAppServlet"**, loaded by **"<system class loader>"**, occupies **193,550,808 (76.29%)** bytes. The memory is accumulated in one instance of **"java.lang.Object[]"** loaded by **"<system class loader>"**.

Keywords
java.lang.Object[]
com.ibm.issf.atjolin.badapp.BadAppServlet

Details »

__ k. Click the **Details** link to see the Shortest path to the Accumulation Point.

Shortest Paths To the Accumulation Point

Class Name	Shallow Heap	Retained Heap
java.lang.Object[256] @ 0xf179788	1,040	193,550,648
java.util.ArrayList @ 0x419bbe8	24	193,550,672
class com.ibm.issf.atjolin.badapp.BadAppServlet @ 0x3d4f388 **System Class**	**80**	**193,550,808**

Accumulated Objects

Class Name	Shallow Heap	Retained Heap	Percentage
class com.ibm.issf.atjolin.badapp.BadAppServlet @ 0x3d4f388	80	193,550,808	76.29%
java.util.ArrayList @ 0x419bbe8	24	193,550,672	76.29%
java.lang.Object[256] @ 0xf179788	1,040	193,550,648	76.29%
com.ibm.issf.atjolin.badapp.BadAppServlet$C @ 0x21fe988	24	1,024,072	0.40%

A memory **accumulation point** is where the memory of many small objects is accumulated under one object. Accumulation points are identified from the large difference in the retained size of an object and the retained size of its child objects.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

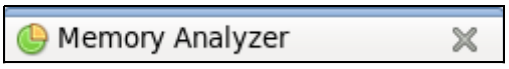
Information

Administrators would have enough information to determine that the cause of OutOfMemory exceptions they are seeing in their log files is mostly likely to be the application that is installed on their server. The next step would be for the administrator to communicate the results of the Memory Analyzer analysis to the application developers. Then, the developers must begin reviewing their code to determine what is causing the data structures to leak memory.

__ 8. **Optional**: Import the other heap dump that you generated, `heapdump_beforeOOM.phd`, into the Memory Analyzer and examine the analysis of the heap before the OutOfMemory exception was thrown. This step might be necessary if the analysis of the heap dump that was generated after the OutOfMemory exception did not identify any leak suspects.

__ 9. When you are done exploring the analysis, close the Memory Analyzer tab.

__ a. From the IBM Support Assistant, click the **Close** icon on the **Memory Analyzer** tab.



__ b. Stop IBM Support Assistant from a terminal window by using the command:

```
/opt/IBM/ISA/ISA5/stop_isa.sh
```

Troubleshooting

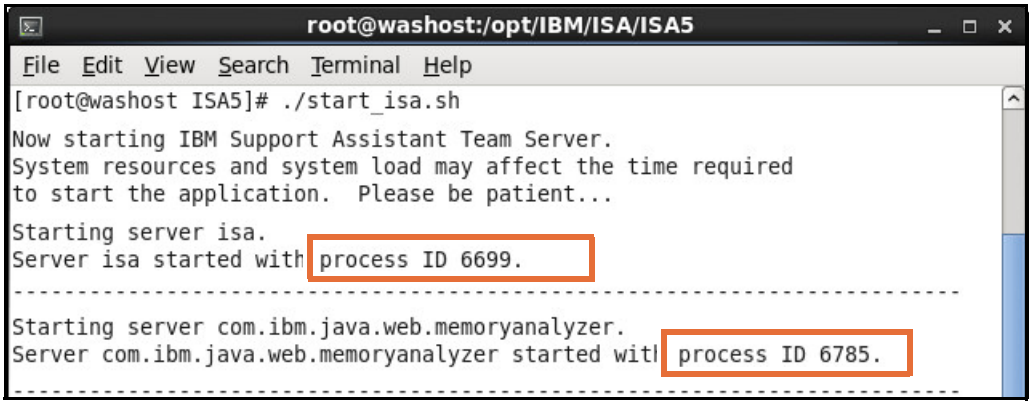
If the Memory Analyzer web edition and IBM Support Assistant do not stop after a few minutes, their processes are probably hung.

Manually stop the Memory Analyzer process by using the command:

`kill -9 <PID>`, where PID is the process ID for the Memory Analyzer [Web].

Manually stop the IBM Support Assistant process by using the command:

`kill -9 <PID>`, where PID is the process ID for IBM Support Assistant.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

You can find the process IDs for the Memory Analyzer [Web] server and the IBM Support Assistant server in the terminal window where you ran the `start_isa.sh` command.

Section 3: Use the Memory Analyzer to view the system core

The system core file that is generated automatically when an OutOfMemory error is thrown is the most complete diagnostic data to use for analyzing the cause of the problem. The Memory Analyzer is able to analyze systems cores and also heap dumps.

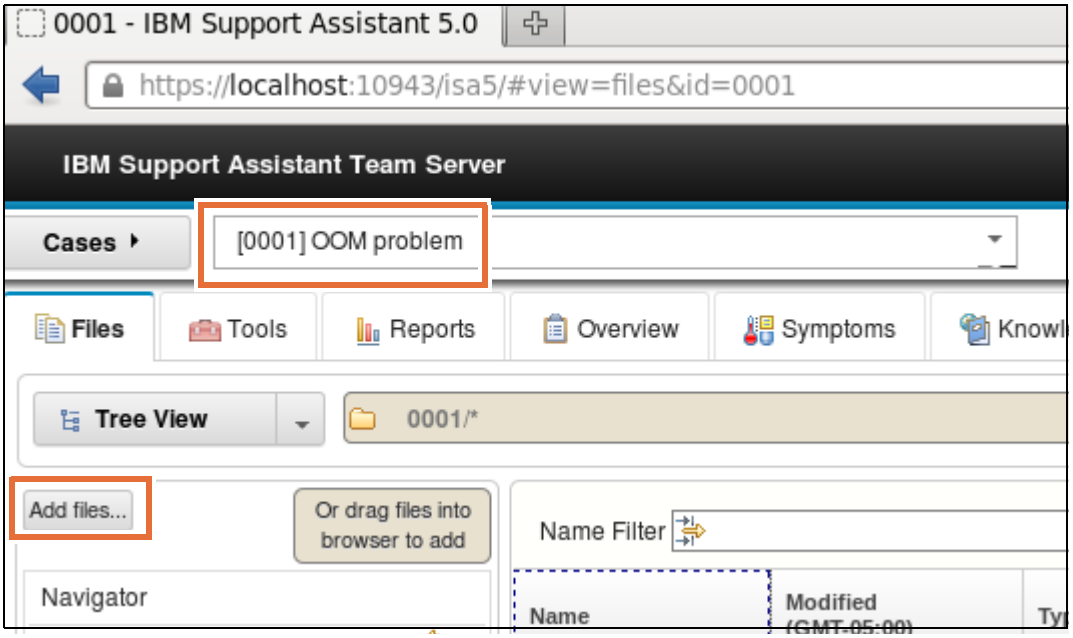
__ 1. Start IBM Support Assistant from a terminal window.

__ a. Enter the command: `/opt/IBM/ISA/ISA5/start_isa.sh`

__ b. Take note of the process ID for the Memory Analyzer server_____

__ 2. Add the system core file to the case.

__ a. Make sure that the case name `OOM problem` shows up beside the Cases button.



__ b. Click the **Files** tab and then click **Add files**.

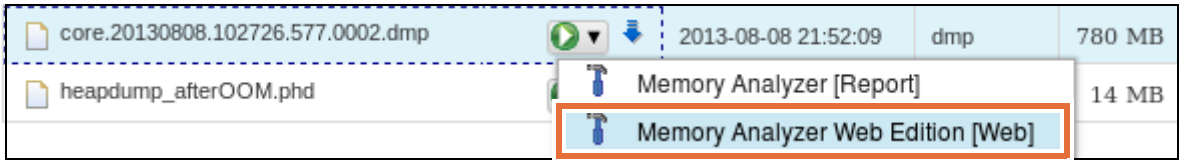
__ c. Navigate to `/usr/Student/OutOfMemoryPD/Dumps` and select the core file.

__ d. Click **Upload**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 3. Analyze the system core file with the Memory Analyzer.

__ a. As soon as the core file is added to the OOM problem case, you can see it listed along with the `heapdump_afterOOM.phd` file in the Files tab.

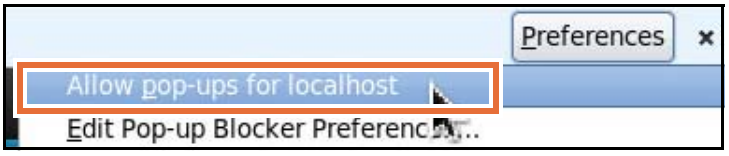


__ b. Click the Quick Tool Launcher icon to see a list of tools. Click the **Memory Analyzer Web Edition**.

__ c. In the Run Tool window, click **Launch**.

__ d. The Memory Analyzer starts up in a web browser. You might have to allow pop-ups in your browser before the processing begins.

__ e. Click **Preferences** and then click **Allow pop-ups for localhost**.



__ f. On the “This Connection is Untrusted” window, click “**I understand the risks**.”

__ g. Click **Add Exception**.

__ h. Click **Confirm Security Exception**.

__ i. The Memory Analyzer begins parsing the system core. The parsing can take about 3 minutes to complete.

__ j. On the Getting Started pane, select **Leak Suspects Report**, and click **Finish**.

__ k. The Leak Suspects Report that is based on the analysis of the system core file typically looks similar to the report that was based on the heap dump file. Examine the **Overview** tab and compare it to what you saw in the heap dump report.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ l. Examine the Leak Suspect information.

Problem Suspect 1

The class **"com.ibm.issf.atjolin.badapp.BadAppServlet"**, loaded by **"com.ibm.ws.classloader.CompoundClassLoader @ 0x3af90b0"**, occupies **199,699,372 (72.15%)** bytes. The memory is accumulated in one instance of **"java.lang.Object[]"** loaded by **"com.ibm.oti.vm.BootstrapClassLoader @ 0x20d9218"**.

Keywords
com.ibm.oti.vm.BootstrapClassLoader @ 0x20d9218
java.lang.Object[]
com.ibm.issf.atjolin.badapp.BadAppServlet
com.ibm.ws.classloader.CompoundClassLoader @ 0x3af90b0

Details »

__ m. Notice that the name of the specific Classloader for the BadAppServlet is now reported based on the more detailed data that is in the system core.

Information

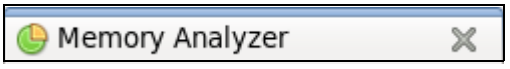
System core versus heap dump data

The system dump provides much data that is not available in the heap dump. In a system dump, you can use the Memory Analyzer to examine:

- Field names
- Primitive fields
- Primitive array contents
- Accurate GC roots
- Native memory
- Native threads

__ 4. When you are done exploring the analysis, close the Memory Analyzer tab.

__ a. From the IBM Support Assistant, click the **Close** icon on the **Memory Analyzer** tab.



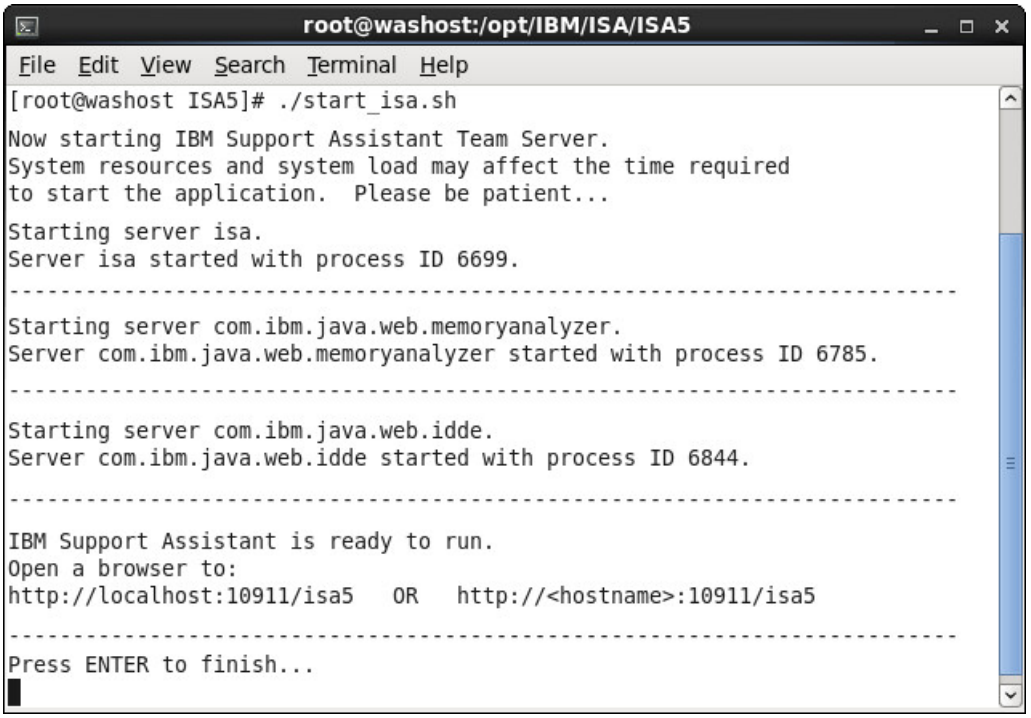
__ 5. Stop the IBM Support Assistant.

__ a. Close the browser; click **File > Quit**.

__ b. Stop the IBM Support Assistant by clicking **Applications** on the desktop. Select **Stop IBM Support Assistant 5**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. From the terminal window, where you started IBM Support Assistant, press **Enter**.



End of exercise

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

Part 1

This exercise showed you how to use Tivoli Performance Viewer to monitor a JVM for out-of-memory conditions. You also learned how to enable verbose GC for an application server, generate a verbose GC log, and use GCMV to interpret the verbose GC log. Then, you configured the WebSphere lightweight memory leak detection mechanism. Finally, you learned how to enable Java heap dumps for an application server, and generate a Java heap dump.

Part 2

In this part, you worked with an application that had a built-in memory leak. You ran the application until it threw an OutOfMemoryError exception. You then examined log files and gathered diagnostic data such as verbose GC and a heap dump. You used tools such as GCMV and Memory Analyzer to analyze the diagnostic data.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 7. Troubleshooting database connection problems

What this exercise is about

This exercise shows how to collect diagnostic data for database connection issues. In the first part, you use various methods to gather diagnostic data. In the second part, you simulate a remote database server outage, and troubleshoot the problem. In the third part, you troubleshoot a data source configuration problem.

What you should be able to do

At the end of this exercise, you should be able to:

- Access and interpret runtime messages in the administrative console
- Test a data source connection to a database by using the Test connection facility in the administrative console
- Examine server logs by using the HPEL Log Viewer
- Locate and examine FFDC logs
- Configure tracing for a data source and interpret the trace data
- Troubleshoot database connection problems

Introduction

Many of the problems that are associated with connecting to an application database are related to configuration of the WebSphere data source object and J2C authentication alias. When there is such misconfiguration, there are several ways to test connections and generate diagnostic data. In the exercise, you gain some experience in troubleshooting database connection issues.

Requirements

HPEL must be configured for server1. HPEL is configured in **Exercise 2: Introduction to problem determination tools**, **Section 2: Enable and configure HPEL**. Two Jython scripts are required, and they are contained in the Student directory on the course VMware image.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

7.1. Simulate a database connection problem and collect diagnostic data.

In this part of the exercise, you explore different methods for collecting diagnostic data about database connection issues.

Note

<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles

Section 1: Preparing the environment

You make a configuration change that results in database connection errors.

__ 1. Ensure that the deployment manager is started.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./startManager.sh`

__ 2. Ensure that the node agent is started and server1 is stopped.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./startNode.sh`

__ c. If server1 is started, stop it with:

```
./stopServer.sh server1 -user wasadmin -password web1sphere
```

__ 3. Rename the existing database that is associated with the PlantsByWebSphere data source.

__ a. Log on to the administrative console as follows:

- User name: `wasadmin`
- Password: `web1sphere`

__ b. Navigate to **Resources > JDBC > Data sources**.

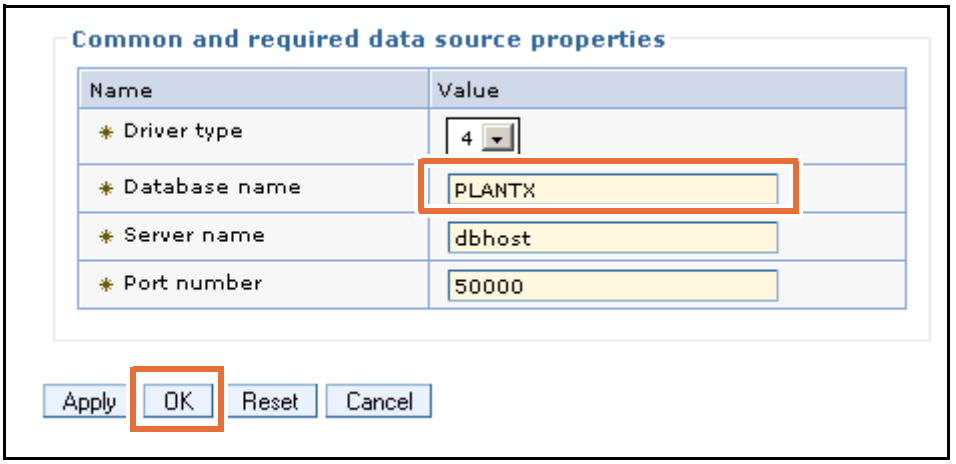
__ c. Click the **Plants** link.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Scroll down on the configuration tab to the data source properties section.

__ e. Change the **Database name** to: PLANTX



__ f. Click **OK**.

__ g. Notice the server name, **dbhost**, and the port number, **50000**. You might need this information later in this exercise.

__ h. Click the link to **Save** changes directly to the master configuration.

__ i. Wait for the nodes to synchronize and click **OK**.

__ 4. Start **server1** from the administrative console.

__ a. Select **Servers > Server Types > WebSphere application servers**.

__ b. Select the check box for **server1** and click **Start**.

__ c. Wait for server1 to start. If you see a message that says that the server failed to start in the required amount of time, wait a few seconds and click the status refresh icon.

Section 2: Investigate the symptoms of the misconfigured data source

In this section, you examine some of the symptoms that result when the data source is misconfigured. In particular, you look at the error messages that are written to various logs.

__ 1. Try accessing the PlantsByWebSphere application.

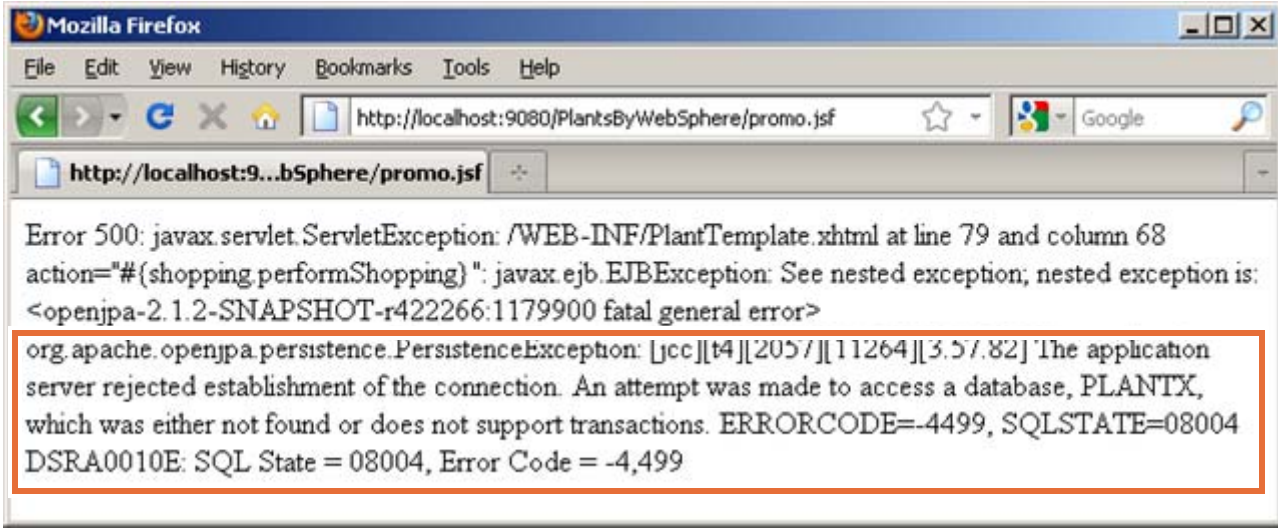
__ a. Start a web browser and enter the web address:

`http://localhost:9080/PlantsByWebSphere`

__ b. Click any tab on the Welcome page, such as **Trees**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Examine the error message that is shown on the browser page.



__ d. Notice that the application server was not able to establish a connection with the database, PLANTX. The reason that is given is that the database was not found or does not support transactions.

Information

DSRA0010E error code explanation

DSRA error messages come from the JCA Resource adapter. A database error occurred for which the SQL state and the Error code are listed. You must consult your database software documentation for more details about SQL State=08004, and Error code=-4,499. Typically this code means that the database name is not a database name that the database server recognizes.

See the IBM technote: *How to interpret SQLException due to "The application server rejected establishment of the connection"*:

`http://www.ibm.com/support/docview.wss?uid=swg21270638`

__ 2. Check the runtime messages in the administrative console.

__ a. Click **Troubleshooting > Runtime Messages > Runtime error**.

Note

Runtime event level

The runtime event level was set to **Info** in a previous exercise and that level shows all messages. If your level is still set to **None**, perform the following steps:

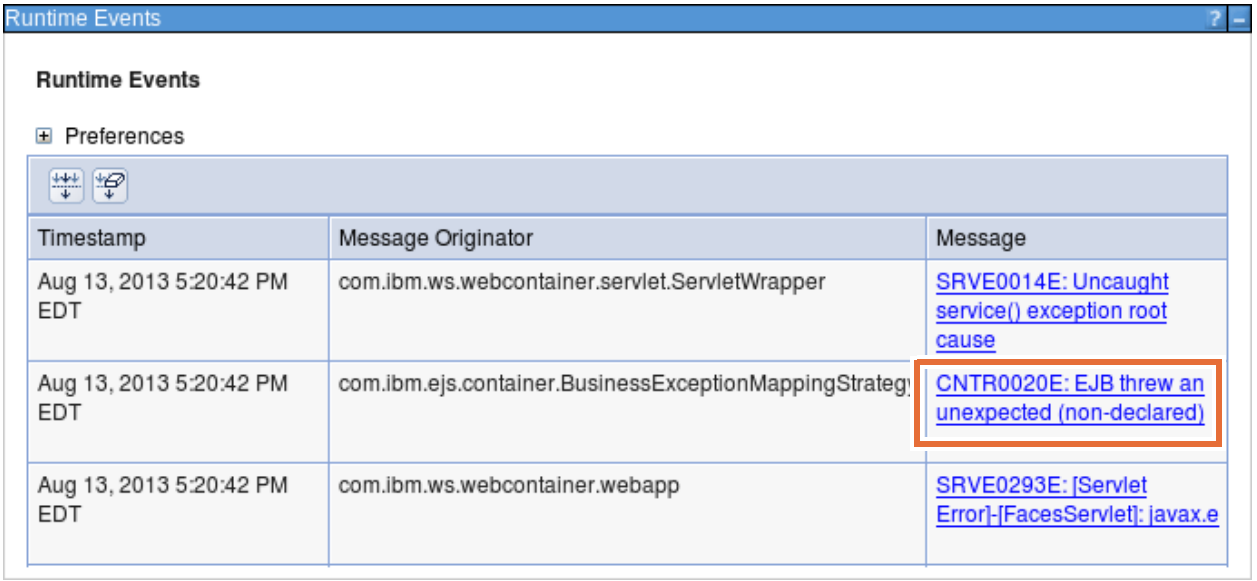
- Select the **Info** event level from the menu, and click **Apply**.
- **Save** changes to the configuration.
- Restart **server1**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

- Go back to **Step 1** and reproduce the problem.

__ b. You might see several messages that are listed. Starting with the most recent, look for messages that relate to the database connection problem. If you see the message code **CNTR0020E**, click it.



__ c. A pane opens that provides details about the message. Explore the details. You see a long stack trace and below the trace, Explanation, and User Action sections

__ d. In the Message detail section, notice the message codes.

```
CNTR0020E: EJB threw an unexpected (non-declared) exception during invocation
of method "getItemsByCategory" on bean
"BeanId(PlantsByWebSphere#PlantsByWebSphereWeb.war#CatalogMgr, null)".
 Exception data: <openjpa-2.1.2-SNAPSHOT-r422266:1179900 fatal general error>
org.apache.openjpa.persistence.PersistenceException:
[jcc][t4][2057][11264][3.57.82] The application server rejected establishment of the connection.
An attempt was made to access a database, PLANTX, which was either not found
or does not support transactions. ERRORCODE=-4499, SQLSTATE=08004
DSRA0010E: SQL State = 08004, Error Code = -4,499 at
org.apache.openjpa.jdbc.sql.DBDictionaryFactory.newDBDictionary(DBDictionaryFactory.java:102)
```

__ e. The first exception was thrown during an attempt to invoke a method that needed access to the PLANTX database.

Information

CNTR0020E error code explanation

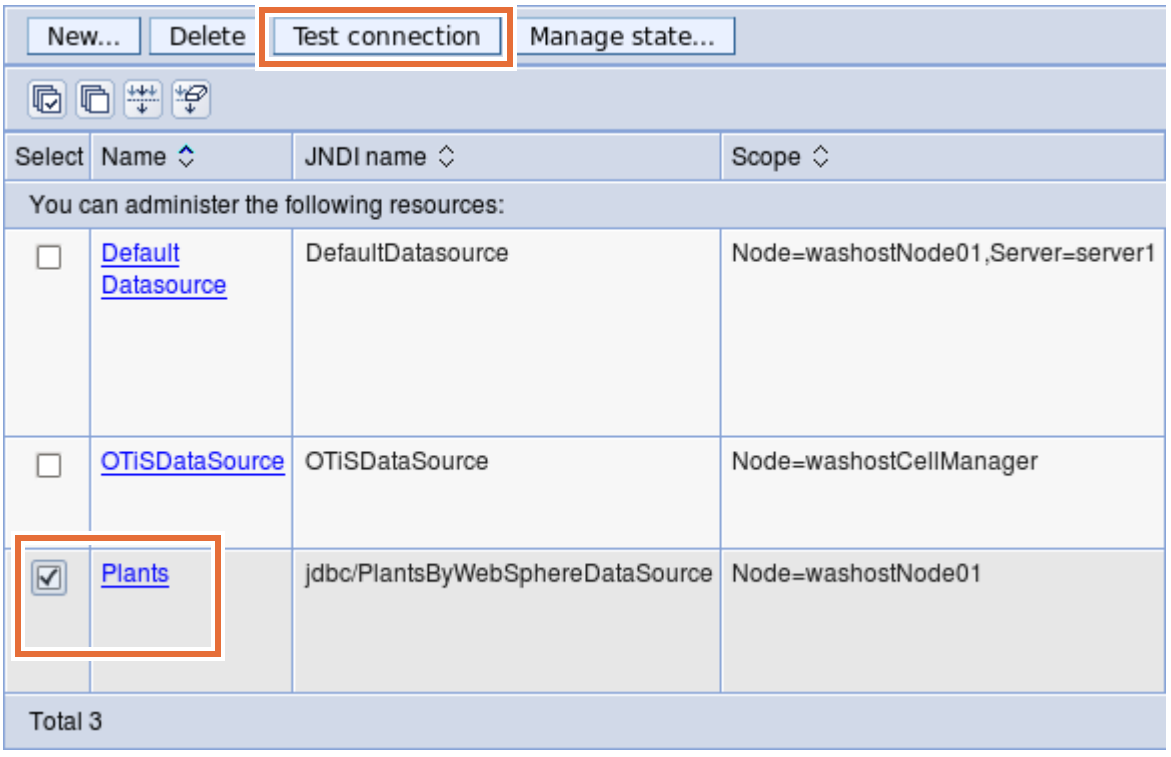
CNTR messages come from the EJB container. An exception, which is not declared on the EJB method signature, was created when the EJB container attempted to invoke this EJB method.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

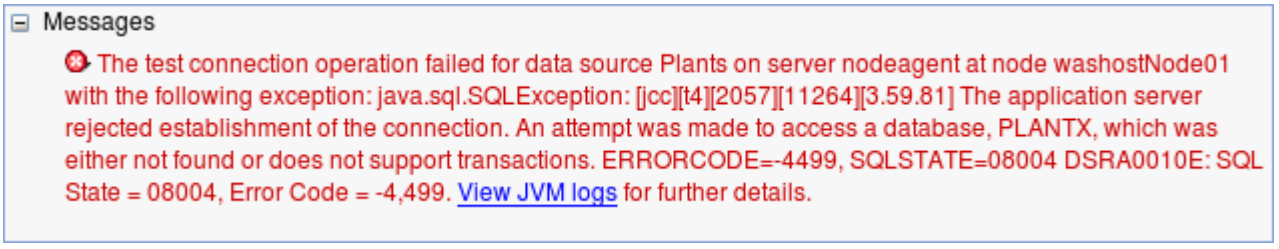
__ 3. Try verifying the data source by using the Test connection button.

__ a. In the administrative console, navigate to **Resources > JDBC > Data sources**.

__ b. Select the **Plants** data source, and click **Test connection.**



__ c. Notice the connection failure message in the console.



__ d. It is basically the same message that you saw in the runtime events and in the web browser. This message directs you to view the JVM logs for more details.

__ 4. Examine the JVM logs.

Note

Enable HPEL for server1

This exercise assumes that you finished **Exercise 2**, and HPEL is enabled. If HPEL is not enabled, perform the following steps:

1) From the administrative console, click **Troubleshooting > Logs and trace > server1**.

2) Click **Switch to HPEL Mode**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3) Click **Save** to save the configuration.

4) Restart **server1**.

__ a. From the administrative console, click **Troubleshooting > Logs and trace > server1**.

__ b. Click **View HPEL logs and trace**.

__ c. Use the Log Viewer features to examine the log messages. Click **Last Page**, and search for any of the message codes that you saw in the previous messages such as `DSRA0010E` and `CNTR0020E`.

__ d. Show only selected threads or limit the search for messages at the SEVERE level.

__ e. Locate the FFDC messages. Enter ***FFDC*** in the **Message contents** field**,** and click **Apply**.

__ f. Locate the following message:

```
com.ibm.ws.ffcd.impl.FfcdProvider logIncident FFDC1003I: FFDC Incident emitted on
/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/ffdc/server1 . . .
com.ibm.ws.rsadapter.jdbc.WSJdbcDataSource.getConnection 299
```

com.ibm.ws.ffdc.impl.FfdcProvider logIncident FFDC1003I: FFDC Incident emitted on /opt/IBM/WebSphere/AppServer/profiles/profile1/logs/ffdc/server1_fc9ad768_13.08.13_17.20.01.6575957608692030787 83.txt com.ibm.ws.rsadapter.jdbc.WSJdbcDataSource.getConnection 299

__ g. Locate the text file in the file system where the FFDC incident is recorded, and examine its contents with a text editor. You also can click the hyperlink and the FFDC text file opens in a viewer.

Information

In this example exercise, the following message was written to the FFDC log.

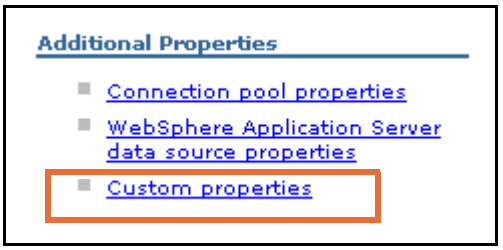
```
FFDC Exception:javax.resource.spi.ResourceAllocationException
SourceId:com.ibm.ws.rsadapter.jdbc.WSJdbcDataSource.getConnection
ProbeId:299 Reporter:com.ibm.ws.rsadapter.jdbc.WSJdbcDataSource@207b72a
javax.resource.spi.ResourceAllocationException: DSRA8100E: Unable to get
a XAConnection from the DataSource.
at
com.ibm.ejs.j2c.FreePool.createManagedConnectionWithMCWrapper(FreePool.ja
va:2359)...
[Stack trace truncated.]
```

__ 5. Finally, configure the data source trace facility, and reproduce the problem.

__ a. In the administrative console, navigate to **Resources > JDBC > Data sources > Plants.**

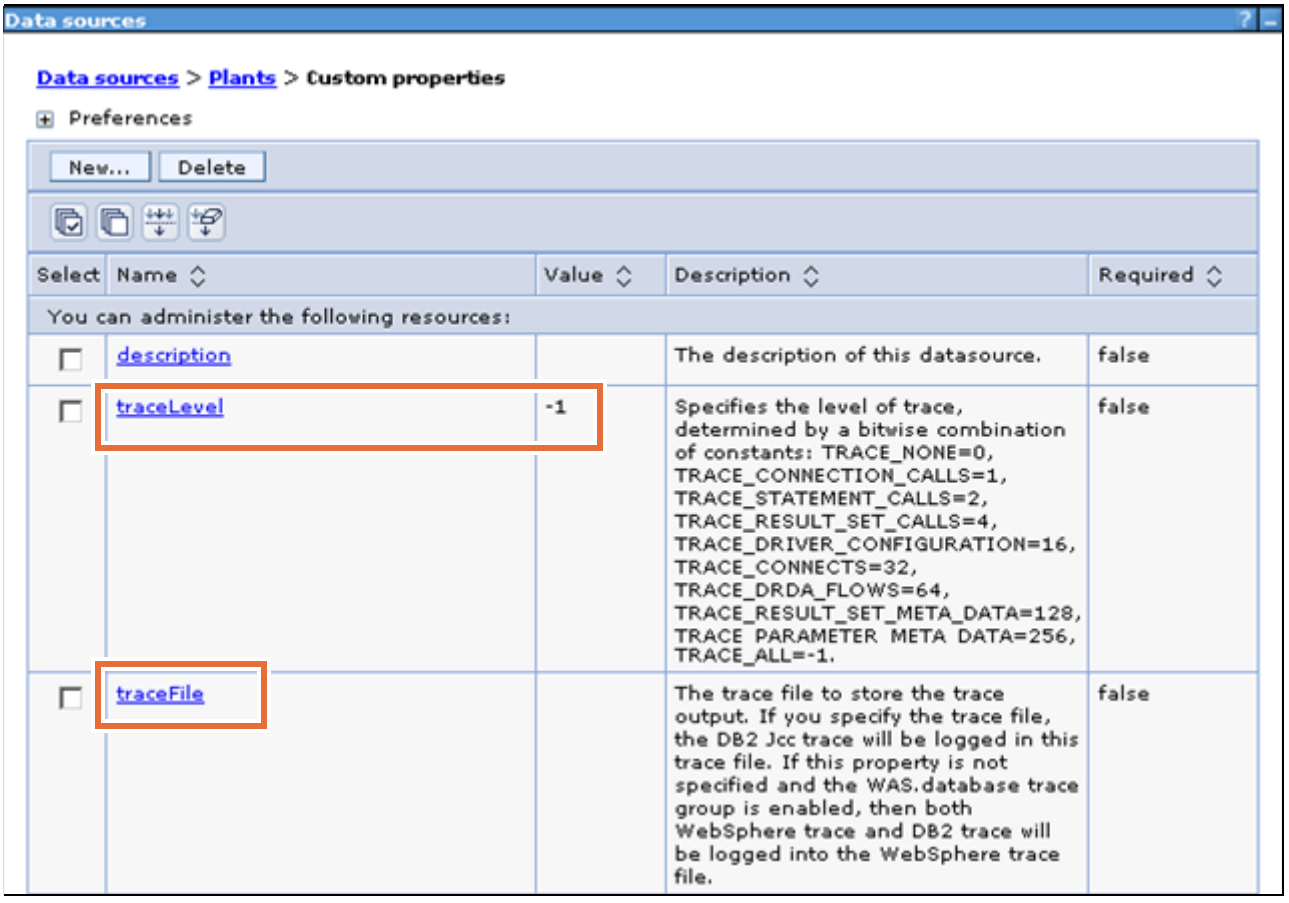
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Under the **Additional properties** section, click **Custom properties**.



Several custom properties can be configured for a data source. For this exercise, you configure the **traceLevel** and the **traceFile** properties.

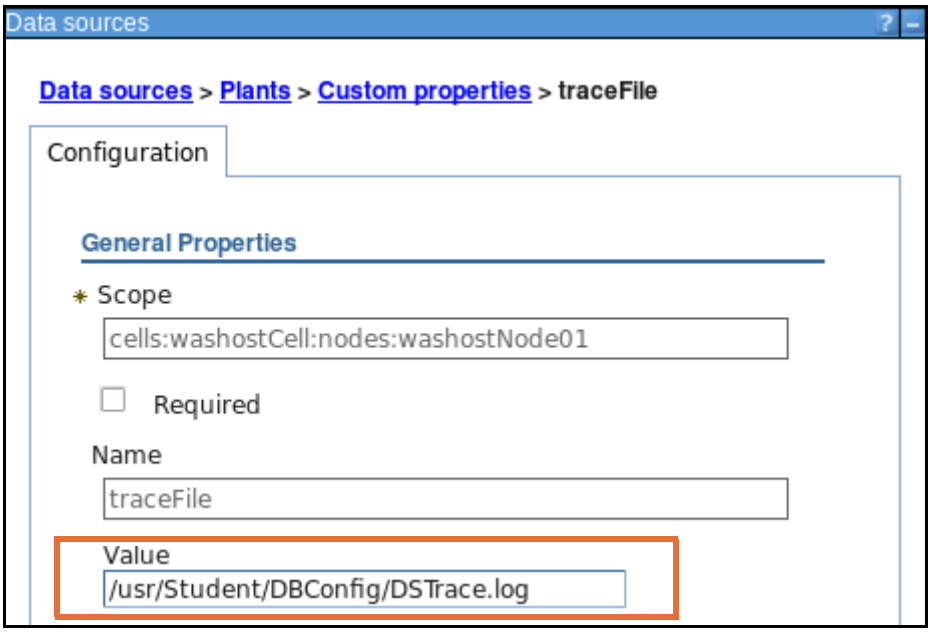
__ c. Accept the default value for traceLevel, **-1**. This value causes all events to be logged. Click the **traceFile** link.



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Enter the value for the traceFile as: `/usr/Student/DBConfig/DSTrace.log`



__ e. Click **OK**.

__ f. Click the link to **Save** changes directly to the master configuration.

__ g. Wait for the nodes to synchronize and click **OK**.

__ h. Restart **server1**.

__ 6. View the `/usr/Student/DBConfig/DSTrace.log` file.

__ a. Use a file browser to go to the `/usr/Student/DBConfig` directory.

__ b. Double-click **DSTrace.log** to open it with gedit.

__ c. Examine the log, and you notice a section that is labeled BEGIN TRACE_DIAGNOSTICS. Scroll down and examine all the messages until you see the END TRACE_DIAGNOSTICS. As you recall from the FFDC log, the server tries to

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

connect with the PLANTX database on startup. The failure of that connection and a stack trace are logged in this trace file.

```
[jcc] BEGIN TRACE_DIAGNOSTICS
[jcc][Thread:server.startup : 2][SQLException@201b24d] java.sql.SQLException
[jcc][Thread:server.startup : 2][SQLException@201b24d] SQL state  = 08004
[jcc][Thread:server.startup : 2][SQLException@201b24d] Error code = -4499
[jcc][Thread:server.startup : 2][SQLException@201b24d] Message    =
[jcc][t4][2057][11264][3.57.82] The application server rejected establishment of the
connection.
An attempt was made to access a database, PLANTX, which was either not found or
does not support transactions. ERRORCODE=-4499, SQLSTATE=08004
[jcc][Thread:server.startup : 2][SQLException@201b24d] Stack trace follows
com.ibm.db2.jcc.am.yn: [jcc][t4][2057][11264][3.57.82] The application server rejected
establishment of the connection.
An attempt was made to access a database, PLANTX, which was either not found or
does not support transactions. ERRORCODE=-4499, SQLSTATE=08004
            at com.ibm.db2.jcc.am.bd.a(bd.java:319)
            at com.ibm.db2.jcc.am.bd.a(bd.java:365)
...
```

__ d. Notice that the server startup threads are throwing an SQL exception.

__ e. Close the log file after you are done examining it.

__ 7. Reproduce the problem by accessing the PlantsByWebSphere application.

__ a. Start a web browser and enter the web address:

`http://localhost:9080/PlantsByWebSphere`

__ b. Click any tab on the Welcome page, such as **Flowers**.

__ 8. Again, view the `/usr/Student/DBConfig/DSTrace.log` file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Note

Notice that the trace log `DSTrace.log` is overwritten, rather than appended to. This behavior is configurable by using the data source custom properties. For this exercise, the overwriting behavior makes it easier to compare two different events: server startup versus PlantsByWebSphere access.

__ a. Locate the `BEGIN TRACE_DIAGNOSTICS` section.

```
[jcc] BEGIN TRACE_DIAGNOSTICS
[jcc][Thread:WebContainer : 0][SQLException@203388e] java.sql.SQLException
[jcc][Thread:WebContainer : 0][SQLException@203388e] SQL state  = 08004
[jcc][Thread:WebContainer : 0][SQLException@203388e] Error code = -4499
[jcc][Thread:WebContainer : 0][SQLException@203388e] Message    =
[jcc][t4][2057][11264][3.57.82] The application server rejected establishment of the
connection.
An attempt was made to access a database, PLANTX, which was either not found or does
not support transactions. ERRORCODE=-4499, SQLSTATE=08004
[jcc][Thread:WebContainer : 0][SQLException@203388e] Stack trace follows
com.ibm.db2.jcc.am.yn: [jcc][t4][2057][11264][3.57.82] The application server rejected
establishment of the connection.
An attempt was made to access a database, PLANTX, which was either not found or does
not support transactions. ERRORCODE=-4499, SQLSTATE=08004
            at com.ibm.db2.jcc.am.bd.a(bd.java:319)
            at com.ibm.db2.jcc.am.bd.a(bd.java:365)
...
```

__ b. Notice that in this case, the web container thread is throwing the SQL exception.

__ c. Close the log file after you are done examining it.

Information

Summary of this part of the exercise

The objective of this exercise is to show many sources of diagnostic data that is associated with a simple configuration error. In this example, you explored the error messages that were generated because the DB2 database name was misspelled in the data source properties.

__ 9. Fix the problem by modifying the Plants data source so that it uses the correct DB2 database name, **PLANTS**, and restart **server1**.

__ 10. Verify that the PlantsByWebSphere application is working again.

__ 11. Compare the DSTrace.log files for the successful server start and a successful PlantsByWebSphere request. Search for: `BEGIN TRACE_CONNECTS`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

7.2. Troubleshoot problems when connecting to a remote database server

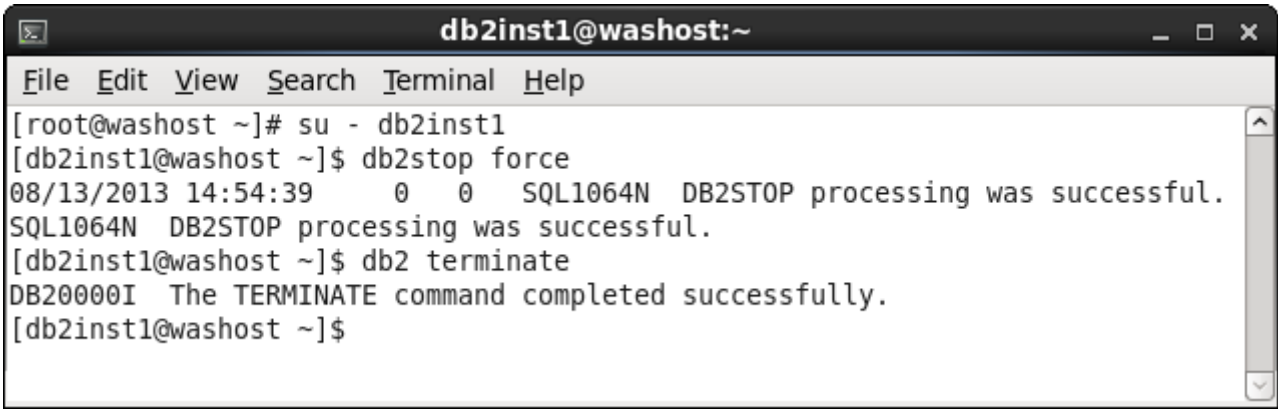
In this section, you simulate problems that might occur when connecting to a remote database server. The lab machine that is used for these exercises is running DB2 Workgroup Server. This server hosts the Plants database that the PlantsByWebSphere application uses. You can stop this server and explore the error messages that show symptoms of the problem.

__ 1. Stop the DB2 instance.

__ a. Open a Terminal window and enter the command: `su - db2inst1`

__ b. Enter the command: `db2stop force`

__ c. Enter the command: `db2 terminate`



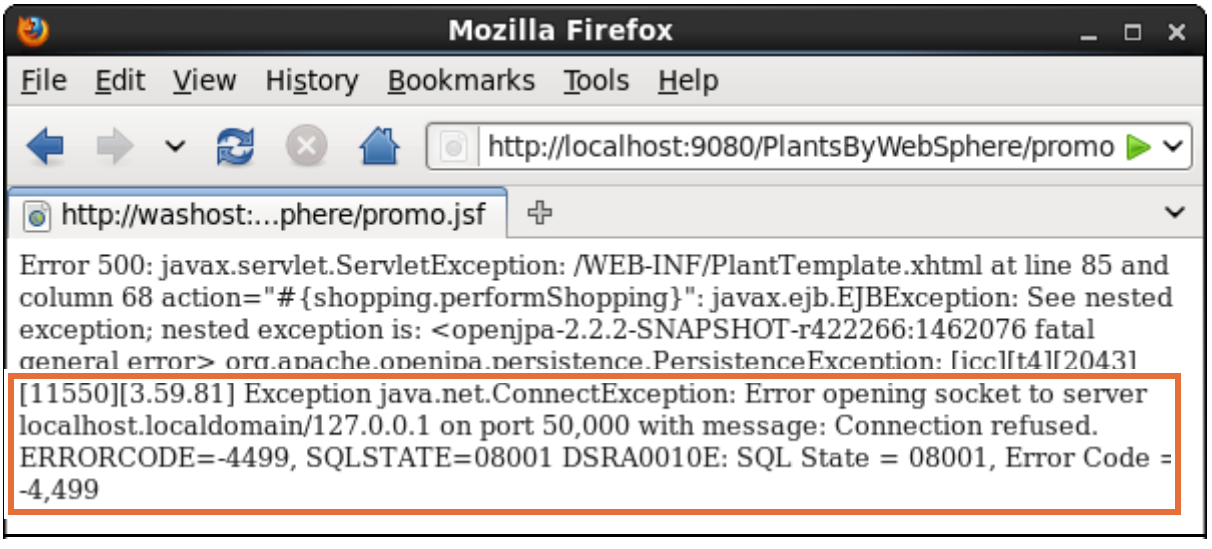
__ 2. Try accessing the PlantsByWebSphere application.

__ a. Start a web browser and enter the web address:

`http://localhost:9080/PlantsByWebSphere`

__ b. Click any tab on the Welcome page, such as **Accessories**.

__ c. Examine the error message that is shown on the browser page.



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Notice the ConnectException message gives specific information about the server where DB2 is running: the host name, IP address is **127.0.0.1** (in a real world scenario the DB2 server would probably be a remote host), and the port on which DB2 listens, **50000**.

Note

Note 1: If you do not see a message with details about the host and IP address, close the browser and repeat steps 2a and 2b.

Note 2: In a real world scenario, the application would catch the connection exception and show a less detailed error page to your customers. Therefore, this valuable troubleshooting data would not be shown. However, you can see similar messages in the trace facility log.

__ 3. View the `/usr/Student/DBConfig/DSTrace.log` file.

__ a. Locate the `BEGIN TRACE_DIAGNOSTICS` section.

```
[jcc] BEGIN TRACE_DIAGNOSTICS
[jcc][Thread:WebContainer : 2][SQLException@1f6c779] java.sql.SQLException
[jcc][Thread:WebContainer : 2][SQLException@1f6c779] SQL state  = 08001
[jcc][Thread:WebContainer : 2][SQLException@1f6c779] Error code = -4499
[jcc][Thread:WebContainer : 2][SQLException@1f6c779] Message    =
[jcc][t4][2043][11550][3.57.82] Exception java.net.ConnectException: Error opening socket
o server dbhost/127.0.0.1 on port 50,000 with message Connection refused: connect.
[jcc][Thread:WebContainer : 2][SQLException@1f6c779] Stack trace follows
com.ibm.db2.jcc.am.yn: [jcc][t4][2043][11550][3.57.82] Exception
java.net.ConnectException: Error opening socket to server dbhost/127.0.0.1 on port 50,000
with message: Connection refused: connect. ERRORCODE=-4499, SQLSTATE=08001
            at com.ibm.db2.jcc.am.bd.a(bd.java:319)
            at com.ibm.db2.jcc.am.bd.a(bd.java:337)
...
```

__ b. Notice that the host name, IP address, and port are also provided in this trace log.

__ 4. Troubleshoot the connection issue. If the connection is refused, and you know the host name, IP address, and port, then you can try various pinging techniques.

__ a. From a command prompt, ping the host name, `dbhost`. In this lab environment, `dbhost` is an alias for the localhost, so the ping is successful. However, in a real world scenario, there might be a problem with DNS resolving the host name, and the ping might fail.

__ b. Try pinging the IP address. Again, the IP 127.0.0.1 is that of the localhost, so this ping is successful. However, in a real world scenario, there might be a network problem, and the ping might fail.

__ c. From a command prompt, enter the following command:

```
netstat -anp
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Scroll down through the output and search for the port `50000`. You do not see a local address 127.0.0.1:50000 in the listening state. From this information, your next step is to verify that the database instance is started on the host.

__ e. Return to the db2 instance window and start the DB2 instance by entering the command: `db2start`

```
db2inst1@washost:~
File  Edit  View  Search  Terminal  Help
[root@washost ~]# su - db2inst1
[db2inst1@washost ~]$ db2stop force
08/13/2013 14:54:39     0   0   SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
[db2inst1@washost ~]$ db2 terminate
DB20000I  The TERMINATE command completed successfully.
[db2inst1@washost ~]$ db2start
08/13/2013 15:04:09     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
[db2inst1@washost ~]$
```

__ f. To log out as `db2inst1`, enter the command: `logout`

__ 5. Verify that the DB2 instance is now available by accessing the PlantsByWebSphere application.

__ a. Start a web browser and enter the web address:

`http://localhost:9080/PlantsByWebSphere`

__ b. Click the **Accessories** tab.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

7.3. Troubleshoot a database connection problem

In this section of the exercise, you run a script that causes some problems that you must troubleshoot and fix with little guidance.

__ 1. Run a script to introduce configuration problems into the Plants data source.

__ a. In a Terminal window, navigate to: `<profiles_root>/profile1/bin`

__ b. Enter the following command, all on one line:

```
./wsadmin.sh -f /usr/Student/DBConfig/BreakDS.py -user wasadmin -password
web1sphere
```

__ c. Wait for the script to complete and the prompt to return.

__ d. Restart **server1** from the administrative console.

__ 2. Discover a symptom of the new problem by trying to access the PlantsByWebSphere application.

__ a. Start a web browser and enter the web address:

`http://localhost:9080/PlantsByWebSphere`

__ b. Click any tab on the Welcome page.

__ c. Examine the error message.

__ 3. Troubleshoot this problem.

__ a. There might be more than one configuration problem, so use an iterative approach to troubleshooting. Test access to PlantsByWebSphere after each correction.

__ b. You can use your instructor as a resource for troubleshooting. In this scenario, your instructor is the database administrator (DBA).

__ c. If you cannot troubleshoot the problem in a reasonable amount of time, run the `FixDS.py` script in the `/usr/Student/DBConfig` directory and restart **server1**. Also, if you open this script with a text editor, you can see how the problem is fixed.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

End of exercise

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In the first part of this exercise, you learned to how to access and interpret runtime messages in the administrative console. You also learned how to test a data source connection to a database by using the Test connection facility. You examined server logs by using the HPEL Log Viewer, and located and examined FFDC logs. You configured tracing for a data source and interpreted the trace data. Finally, you gained some experience with troubleshooting a database connection problem.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 8. Troubleshooting a connection leak

What this exercise is about

This exercise demonstrates the Connection Leak Logic trace facility that is built into WebSphere Application Server and is used to identify application code that is leaking database connections.

What you should be able to do

At the end of this exercise, you should be able to:

- Examine tuning parameters for data source connection pools
- Use a diagnostic provider to examine state data for a data source
- Recognize a tuning parameter problem situation
- Configure tracing and examine the logs

Introduction

WebSphere Application Server includes a lightweight tracing facility to diagnose a connection leak and to pinpoint which application is causing the leak. Previously in WebSphere, J2C tracing was used to detect a leak, but tracing affected performance, and the trace data was not as easy to gather in a production environment. The use of J2C and shared connections in WebSphere Application Server increases the possibility of connection leaks for applications.

Requirements

This exercise uses a simple load tester application to test a sample application, **ConnLeak.ear**, which has a built-in connection leak. For the purposes of this lab, the PlantsByWebSphere data source is used as the back-end database.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Preparing the environment

In this section, you prepare the environment for this exercise by changing the logging mode to Basic, installing the ConnLeak application, and modifying the connection pool parameters of the PlantsByWebSphere data source. You also configure Connection Leak Logic tracing for server1.

__ 1. Ensure that the deployment manager is started.

__ a. From a Terminal window, change the directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./startManager.sh`

__ 2. Ensure that the node agent is running but that server1 is stopped.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./startNode.sh`

__ c. If server1 is started, stop it with: `./stopServer.sh server1 -user wasadmin -password web1sphere`

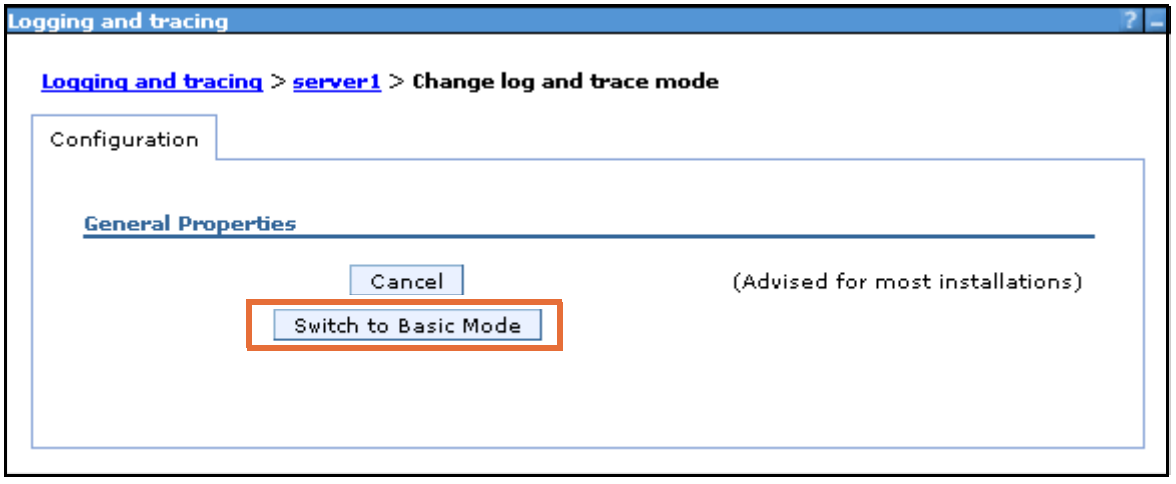
__ 3. Change logging back to Basic mode for server1. Later in this exercise you need to examine a `trace.log` file. The `trace.log` is easier to search for this exercise, so you need to switch to Basic mode.

__ a. Log on to the administrative console as follows:

- User ID: `wasadmin`
- Password: `web1sphere`

__ b. Click **Troubleshooting > Logs and trace > server1 > Change log and trace mode**.

__ c. Click **Switch to Basic Mode**.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Click **Save** to save directly to the master configuration.

__ e. Wait for the node to synchronize and click **OK**.

__ 4. Install the **ConnLeak** application.

__ a. In a Terminal window, change directory to: *<profiles_root>*/profile1/bin

__ b. Enter the following command:

```
./wsadmin.sh -f /usr/Student/ConnLeak/install_ConnLeakApp.py -user
wasadmin -password web1sphere
```

__ c. Wait until you see the following message and for the node synchronization to complete.

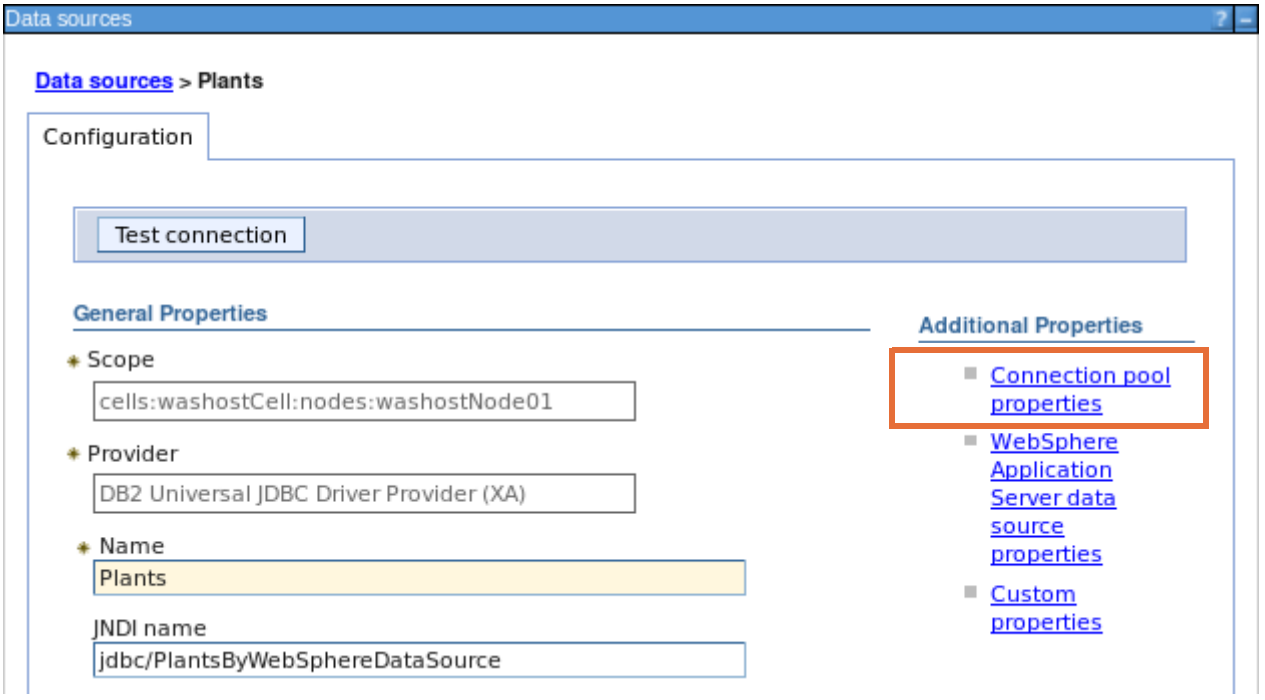
```
ADMA5013I: Application ConnLeak installed successfully.
```

__ 5. The example ConnLeak application uses the PLANTS data source, so you next modify the *Connection timeout* property for the PLANTS data source.

__ a. From the administrative console, click **Resources > JDBC > Data sources**.

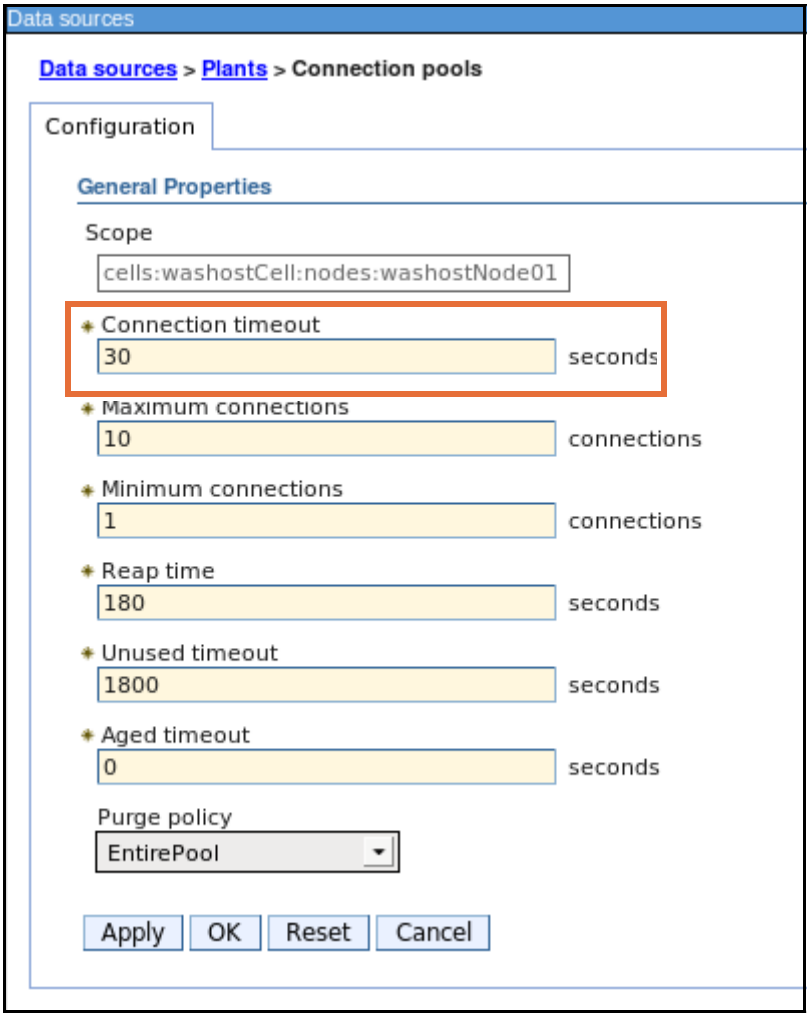
__ b. Ensure that the scope is set to **All scopes**.

__ c. Select **PLANTS > Connection Pool Properties**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Change the **Connection timeout** value to **30**. Changing the value of the Connection timeout to 30 seconds from 180, ensures that the connection requests used in this exercise timeout. The load that is being generated and the connection leak design trigger the Connection Leak logic with this value.



__ e. Notice also that the default value for Maximum connections is `10` and for Minimum connections is `1`. These values are the defaults.

__ f. Click **OK**.

__ g. Click **Save** to save directly to the master configuration.

__ h. Wait for the node to synchronize and click **OK**.

__ 6. Configure Connection Leak Logic tracing on server1.

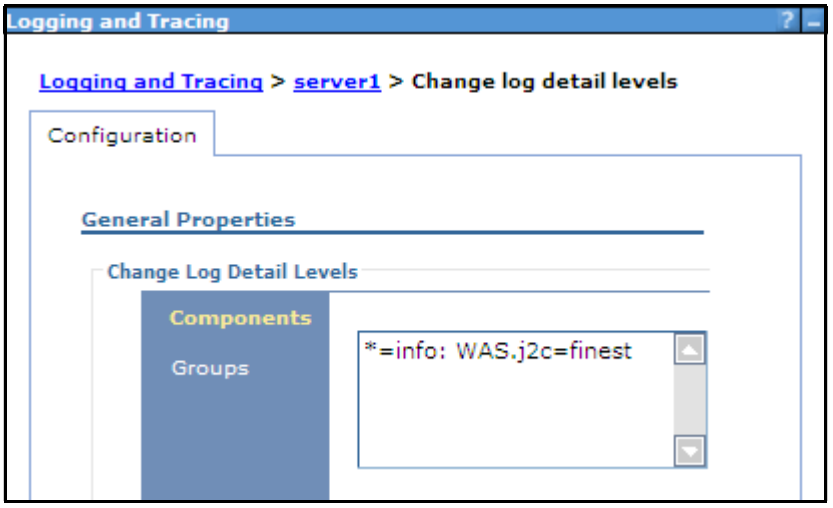
__ a. Using the administrative console, click **Troubleshooting > Logs and trace > server1 > Change log detail levels**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the following string into the trace string field:

```
*=info: WAS.j2c=finest
```



__ c. Click **OK** and **Save** the changes to the master configuration.

__ d. Wait for the node to synchronize and click **OK**.

Section 2: Load testing the application

In this section, you configure the diagnostic provider for the PLANTS data source; then you run a simple load testing application against the ConnLeak application. You examine the state data that is provided and the tracing log to detect symptoms of the connection leak.

__ 1. Start server1.

__ a. In the administrative console, select **Servers > Server Types > WebSphere application servers**.

__ b. Check **server1** and click **Start**.

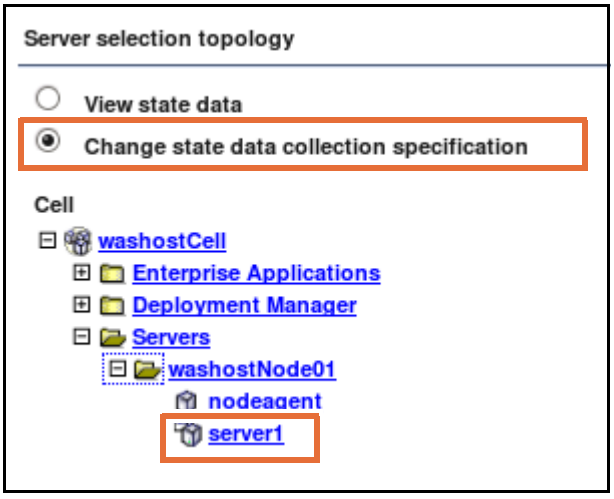
__ c. Wait for the server to start.

__ 2. Configure the diagnostic provider for the PLANTS data source.

__ a. In the administrative console, select **Troubleshooting > Diagnostic Provider > State Data**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

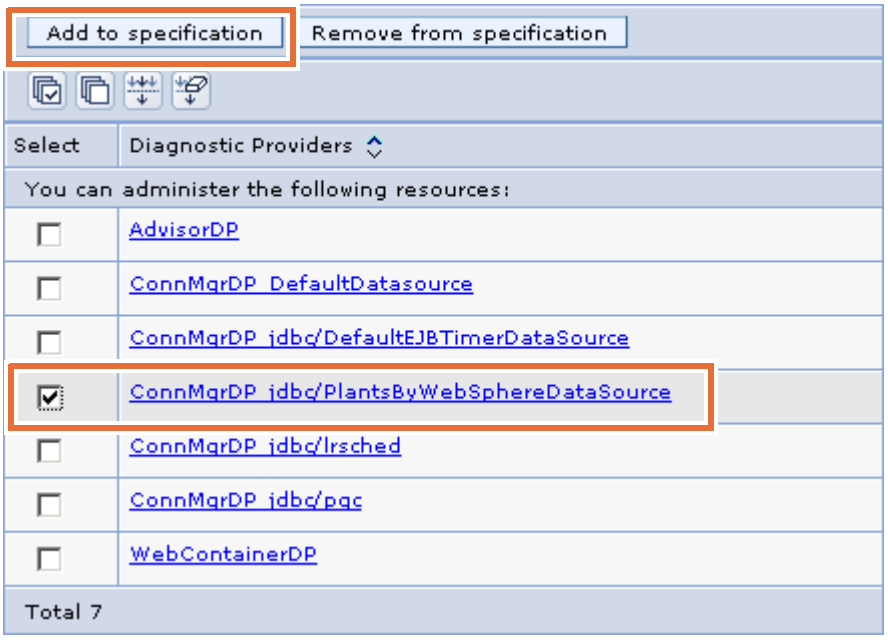
__ b. Under **Server selection topology**, select **Change state data collection specification**, and expand the cell topology until you see **server1**.



__ c. Click **server1**.

__ d. Click the **Runtime** tab in the **Change State Specification** pane.

__ e. Check the box for **ConnMgrDP jdbc/PlantsByWebSphereDataSource**.

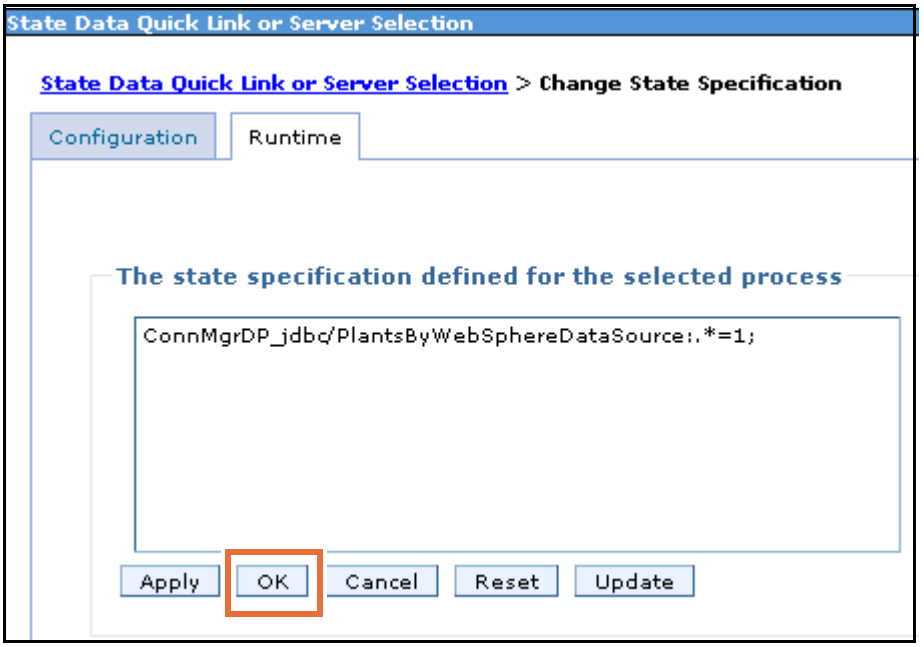


__ f. Click **Add to specification**.

© Copyright IBM Corp. 2013

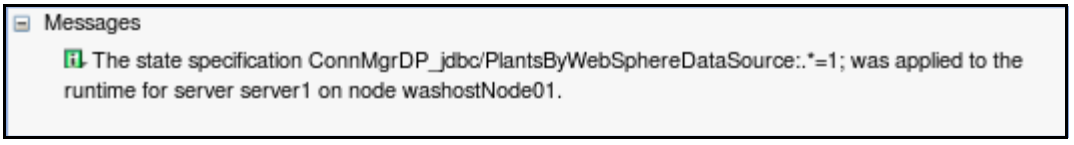
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. Scroll up and verify that the state specification shows up in the window.



__ h. Click **OK**. You then see a message such as:

```
The state specification ConnMgrDP_jdbc/PlantsByWebSphereDataSource:.*=1; was
applied to the runtime for server server1 on node washostNode01.
```



__ i. You are going to view the state data later in this exercise after you have load tested the data source.

__ 3. Use a **LoadTester** application to generate a load against the **ConnLeak** application.

__ a. Open a Terminal window.

__ b. Change directory to: `/usr/Student/ConnLeak`

Note

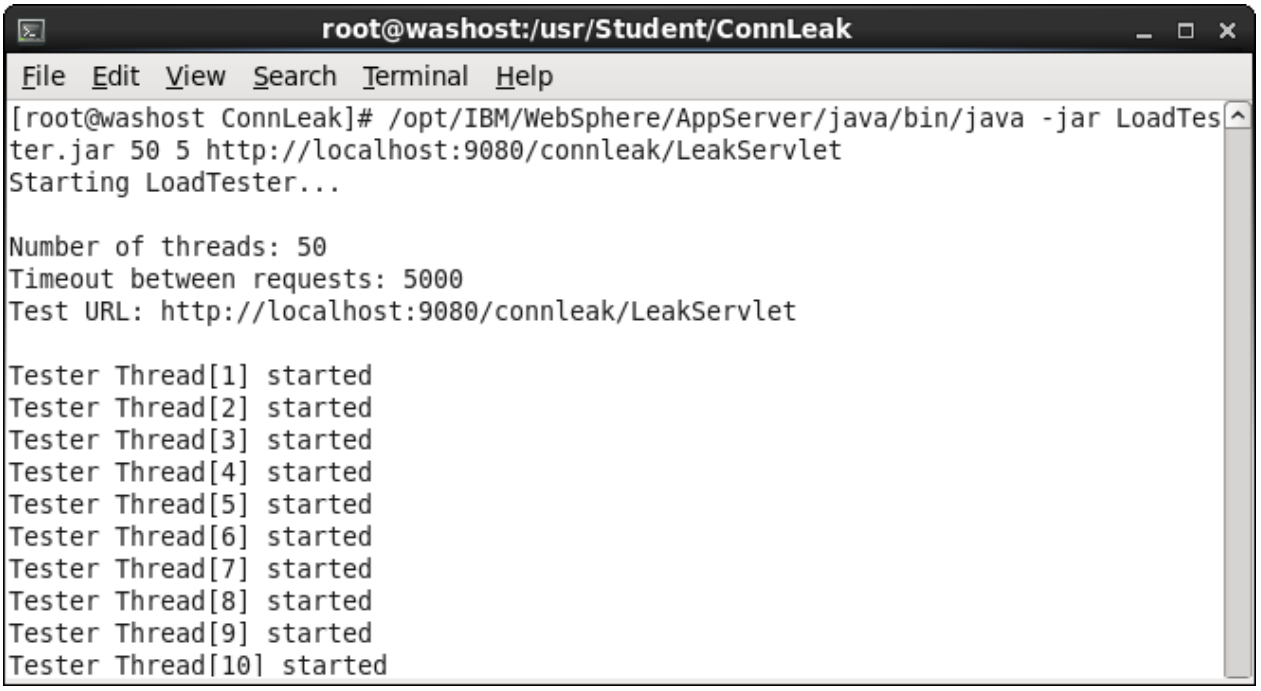
You now run a simple Java program that generates load against the ConnLeak servlet. The command-line format is

```
LoadTester.jar <number of threads> <timeout in seconds> <url>
```

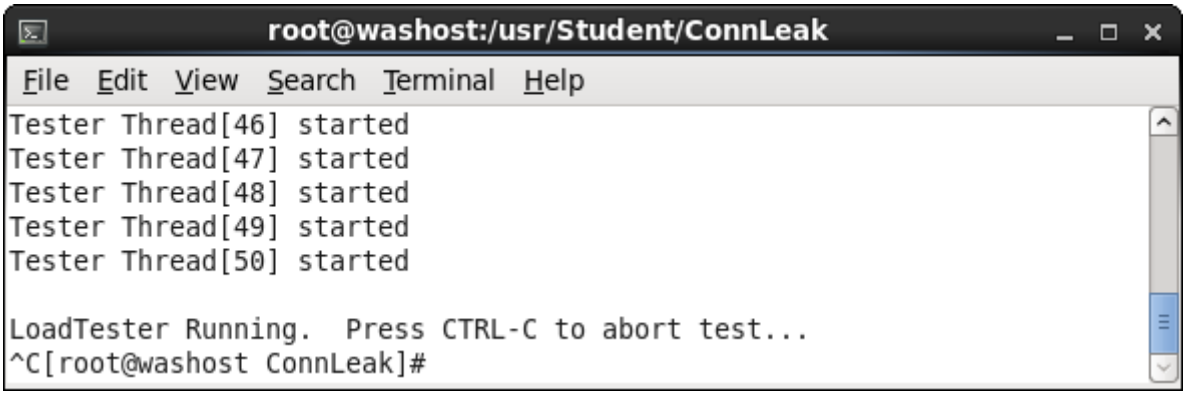

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Enter the following command all on one line:

```
/opt/IBM/WebSphere/AppServer/java/bin/java -jar LoadTester.jar 50 5
http://localhost:9080/connleak/LeakServlet
```



__ d. In the Terminal window, wait for the message `press CTRL-C to abort test...` before proceeding.



__ e. Press **CTRL-C** when prompted in the Terminal window.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

The LoadTester generates 50 threads that simultaneously access the LeakServlet. The small number of JDBC connections, duration of use, and the large number of clients, cause the connection starvation that leads to the connection timeout.

If you examine the `SystemOut.log` for server1 after running the load test, you will see numerous `J2CA0045E` error messages that say:

```
Connection not available while invoking method createOrWaitForConnection for
resource jdbc/PlantsByWebSphereDataSource.
```

This situation is a common problem and is documented in the *IBM technote: How to troubleshoot J2CA0045E connection pooling problems*, `http://www.ibm.com/support/docview.wss?uid=swg21385033`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 3: Determining the cause of the connection leak

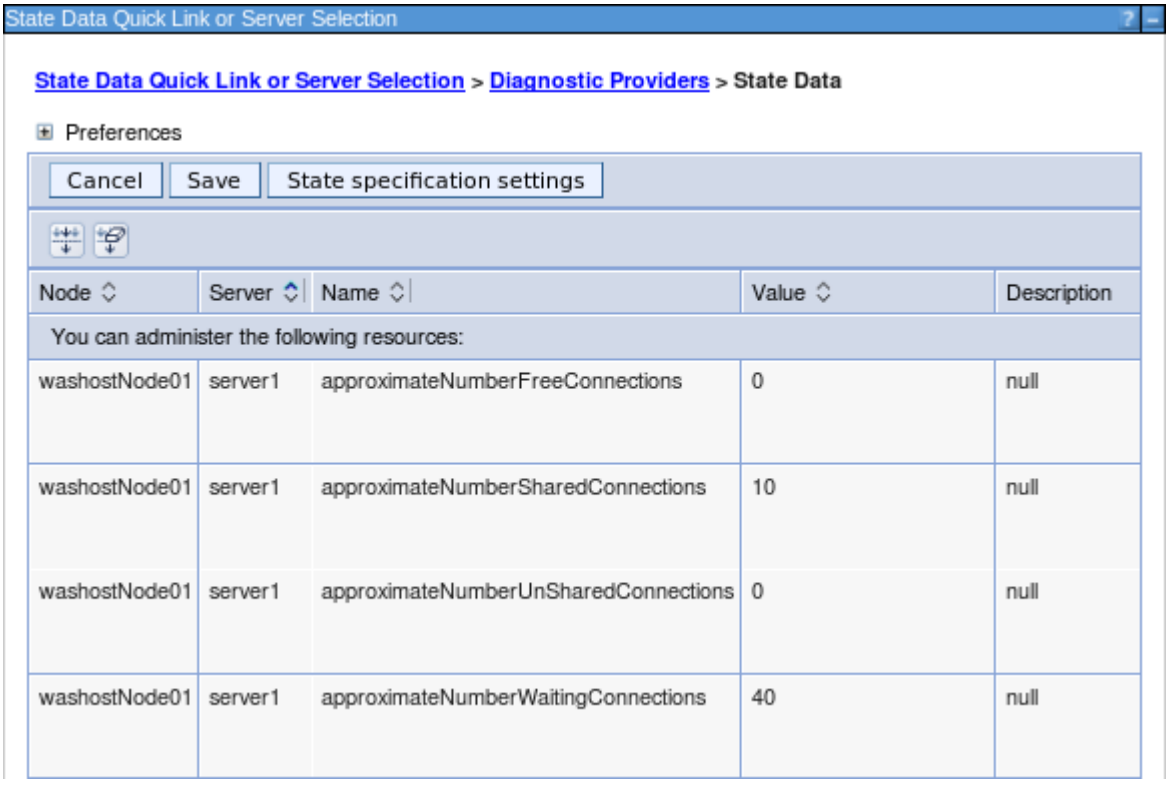
In this section, you examine the state data that is provided for the data source, and the trace log to detect symptoms of the connection leak and troubleshoot the cause of the leak. You also examine the Java code for the LeakServlet to understand the nature of the connection leak.

__ 1. Again, examine the state data for the **PlantsByWebSphereDataSource**.

__ a. From the administrative console, select **Troubleshooting > Diagnostic Provider > State data**.

__ b. Scroll down to the **Server selection topology** section and select **View state data**. Expand the cell topology until you see **server1**.

__ c. Click **server1 > ConnMgrDP_jdbc/PlantsByWebSphereDataSource**. You see a page of state data.



__ d. Notice the number of shared connections. In this example, there are 10, which is also the configured maximum connections. How many waiting connections are there? In this example, there are approximately 40. You might see different numbers. Also, notice that there are several pages of state data. Browse through all the pages, and you can see detailed state data about each connection. **NOTE**: If you see 0 shared connections, run the LoadTester again, and refresh the State Data pane.

__ e. Click **Cancel**.

__ f. Go back to **Section 2, Step 3c** and run the LoadTester application again. View the state data again. Notice the difference in the state data.

__ g. Click **Cancel**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Stop **server1**.

__ a. In the administrative console, select **Servers > Server Types > WebSphere application servers**.

__ b. Check **server1** and click **Stop**.

__ c. Wait for the server to stop; then click **OK**.

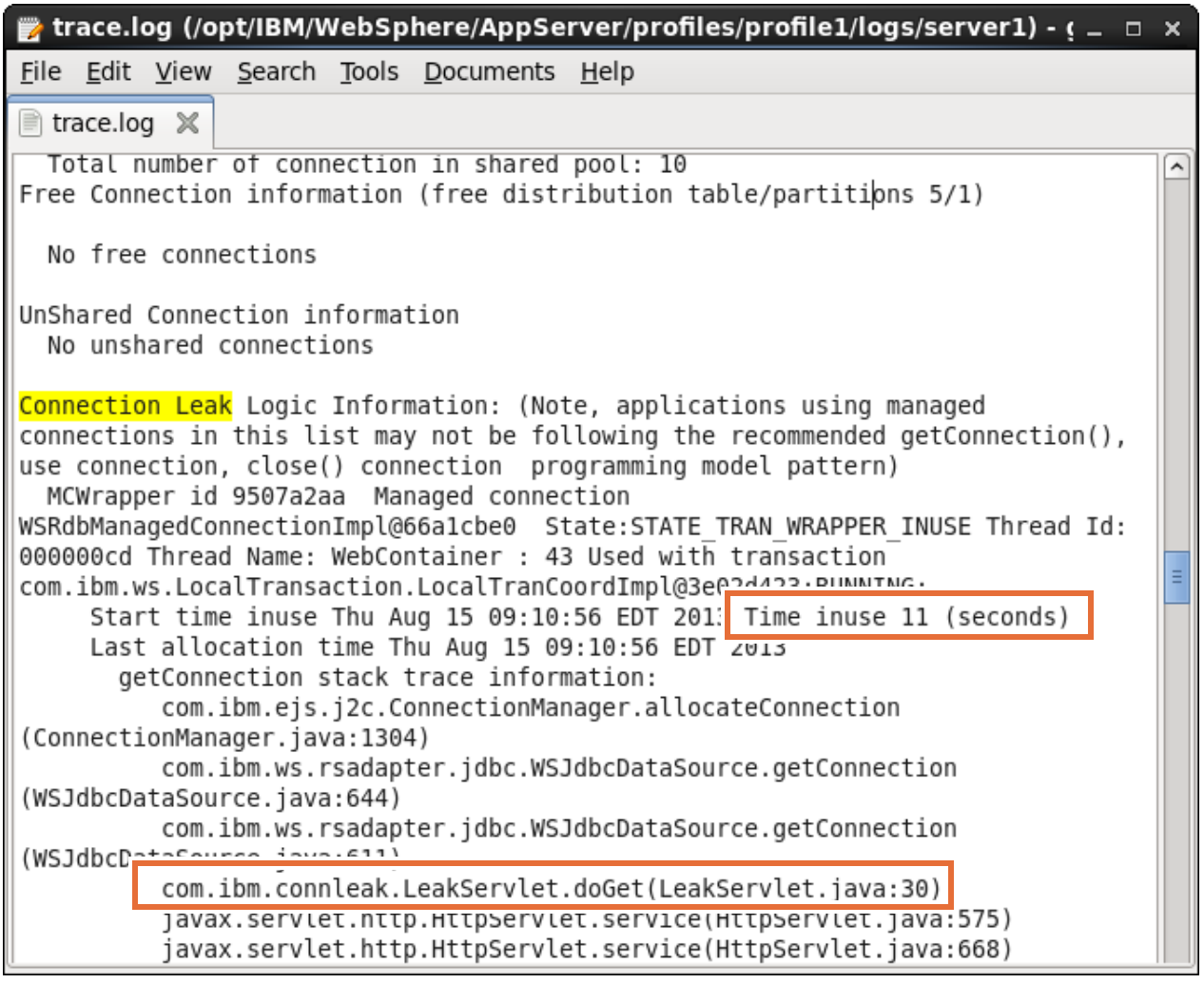
__ 3. Next, you review the trace data that is generated during the test for Connection Leak diagnosis.

__ a. Use a file browser to navigate to: `<profiles_root>/profile1/logs/server1`

__ b. Double-click **trace.log** to open it in gedit.

__ c. Search for the string `Connection Leak` in the trace output. Locate the stack trace immediately following the string. This trace indicates the state of each Managed Connection request.

__ d. In the first stack trace, start at the top and work down until you find the application code that calls the `getConnection` method. The first instance that you find is `com.ibm.connleak.LeakServlet.doGet(LeakServlet.java:30)`. To further diagnose the leak, line 30 of the LeakServlet needs to be analyzed.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Notice that the stack trace points to the LeakServlet as the source of the connection leak.

__ f. Close **trace.log**.

__ 4. Next, you review the application code.

The source for the application is available in the `ConnLeak.ear` file that was installed earlier. That file contains the `ConnLeakWeb.war` file, which contains the `LeakServlet.java` file. To save time, the Java source is provided outside of the application EAR file.

__ a. Using a file browser, navigate to: `/usr/Student/ConnLeak/ErrorSource`

__ b. Right-click **LeakServlet.java** and select **Open with gedit**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Click **Edit > Preference**, and select **Display line numbers**.

LeakServlet.java (/usr/Student/ConnLeak/ErrorSource) - gedit

File Edit View Search Tools Documents Help

LeakServlet.java

```
1 package com.ibm.connleak;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Connection;
6
7 import javax.naming.InitialContext;
8 import javax.servlet.Servlet;
9 import javax.servlet.ServletException;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13 import javax.sql.DataSource;
14
15 public class LeakServlet extends HttpServlet implements Servlet {
16
17         public LeakServlet() {
18                 super();
19         }
20
21         protected void doGet(HttpServletRequest req, HttpServletResponse resp)
   throws ServletException, IOException {
22
23                 PrintWriter pw = resp.getWriter();
24                 pw.println("<HTML><BODY><PRE>");
25                 pw.println("Beginning LeakServlet...");
26
27                 try {
28                 InitialContext ctx = new InitialContext();
29                 DataSource ds = (DataSource) ctx.lookup("java:comp/env/datasource");
30                 Connection conn = ds.getConnection();
31                 conn.close();
32             } catch (Exception e) {
33
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Scroll down to line **30**, where the call to the `getConnection` method is. The next line in the file shows that the connection is being closed after use. Therefore, unclosed connections can be ruled out as the cause of the connection leak.

__ 5. Reviewing the code further, two more methods are called after the connection is closed, `businessMethod1()` and `businessMethod2()`. Both methods sleep for 6 seconds, which prevents the `doGet` method from exiting for at least 12 seconds.

*LeakServlet.java (/usr/Student/ConnLeak/ErrorSource) - gedit

File Edit View Search Tools Documents Help

LeakServlet.java

```
36
37          businessMethod1();
38          businessMethod2();
39
40              pw.println("Completed...");
41                  pw.println("</PRE></BODY></HTML>");
42          }
43
44          private void businessMethod1(){
45                  try {
46                          Thread.sleep(6000);
47                  } catch (InterruptedException e) {
48                          e.printStackTrace();
49                  }
50          }
51
52          private void businessMethod2(){
53                  try {
54                          Thread.sleep(6000);
55                  } catch (InterruptedException e) {
56                          e.printStackTrace();
57                  }
58          }
```

Information

Beginning with WebSphere V5, shared connections are now the default unless otherwise specified. A shared connection remains allocated until the method that calls `getConnection` completes, even if the close method is called. If the parent method continues to make other calls to business methods or EJBs, or forwards control to another servlet, the connection is not available to other threads. This behavior might not be exposed under light load testing, but exhausts the connection pool under a heavy load.

The amount of time that is spent running business methods after the connection is used within the `doGet` method is the cause of the connection leak.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 6. Close `LeakServlet.java`.

Information

Database Connection Pool Analyzer

This tool is available in IBM Support Assistant 4 and employs a heuristic analysis engine to help you resolve problems that are related to Java database Connectivity (JDBC) connection pools and find JDBC connection leaks. The tool analyzes JDBC connection pool manager traces and provides the following functions: analysis of JDBC data source, analysis of JDBC connection pool configuration, JDBC connection chart view, and Java stack trace view of the `getConnection` method.

The tool does the following tasks:

- Parses trace files for the JDBC connection pool manager
- Detects available JDBC data sources and configurations
- Counts the number of waiters of connections and the number of connections
- Finds Java stack traces associated with JDBC connection leaks
- Provides charts of JDBC connections for each JDBC data source

For problems like the one you are analyzing in this exercise, it might be helpful to import the `trace.log` file into the Database Connection Pool Analyzer.

Section 4: Resolving the connection leak

This particular type of connection leak can be resolved in multiple ways. A new EJB can be created that would have a method specifically for data access. The new method would call `getConnection` and retrieve the information that the servlet requests. The connection to the database would be returned to the free pool when the EJB call completes, returning the data to the servlet for further processing. While this option is probably the best solution, modifying application code is not always possible.

Another solution would involve wrapping the `getConnection` method and data access methods with a UserTransaction object. When a global transaction is used, the local transaction that keeps the database connection open until the method completes is suspended. When the global transaction completes, the connection is immediately returned to the free pool. This solution would also require a rewrite of the servlet.

The easiest way to resolve this problem is to use unshared connections instead of shared connections. Unshared connections are immediately returned to the free pool when the close method is called. The behaviors of unshared connections are more like the behavior of WebSphere V4 connection management while using J2C. The type of connection for each resource reference is defined in the deployment descriptor of the application module. The most common way to modify the deployment descriptor is with an EAR file editor like Rational Application Developer or the Application Server Toolkit. In the worst case scenario, the deployment descriptor can be modified by hand and repackaged.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

For the purposes of this exercise, you review the deployment descriptor and compare it against the one from the solution EAR to see the changes that are required to use unshared connections.

Note

Deployment descriptor files for web application modules are named `web.xml`. To save time, the XML files are provided outside of the EAR files for viewing.

__ 1. Open the web deployment descriptor file.

__ a. Using a file browser, navigate to: `/usr/Student/ConnLeak/ErrorSource`

__ b. Open the deployment descriptor file `web.xml`. Right-click the file and open it with gedit.

__ c. Search the `web.xml` file for the string: `res-sharing-scope`
In the `ConnLeak.ear` file, the res-sharing-scope is set to `Shareable`.

```
File  Edit  Format  View  Help
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.4" xmlns="http://java.sun.com/xml
        <display-name>ConnLeakWeb</display-name>
        <servlet>
                <description>
                </description>
                <display-name>
                LeakServlet</display-name>
                <servlet-name>LeakServlet</servlet-name>
                <servlet-class>
                com.ibm.connleak.LeakServlet</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>LeakServlet</servlet-name>
                <url-pattern>/LeakServlet</url-pattern>
        </servlet-mapping>
        <resource-ref id="ResourceRef_1148791860296">
                <description>
                </description>
                <res-ref-name>datasource</res-ref-name>
                <res-type>javax.sql.DataSource</res-type>
                <res-auth>Container</res-auth>
                <res-sharing-scope>Shareable</res-sharing-scope>
        </resource-ref>
</web-app>
```

__ 2. Open the solution web deployment descriptor file.

__ a. Using a file browser, navigate to `/usr/Student/ConnLeak/SolutionSource`.

__ b. Open the deployment descriptor file `web.xml`. Right-click the file and open it with gedit.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. In the `web.xml` file from the solution EAR, you see that the `res-sharing-scope` property is set to `Unshareable`. All other files in the EAR are identical to the original `ConnLeak.ear`.

File Edit Format View Help

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
        <display-name>ConnLeakWeb</display-name>
        <servlet>
                <description>
                </description>
                <display-name>
                LeakServlet</display-name>
                <servlet-name>LeakServlet</servlet-name>
                <servlet-class>com.ibm.connleak.LeakServlet</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>LeakServlet</servlet-name>
                <url-pattern>/LeakServlet</url-pattern>
        </servlet-mapping>
        <resource-ref id="ResourceRef_1148791860296">
                <description>
                </description>
                <res-ref-name>datasource</res-ref-name>
                <res-type>javax.sql.DataSource</res-type>
                <res-auth>Container</res-auth>
                <res-sharing-scope>Unshareable</res-sharing-scope>
        </resource-ref>
</web-app>
```

__ d. After reviewing the changes, close both files.

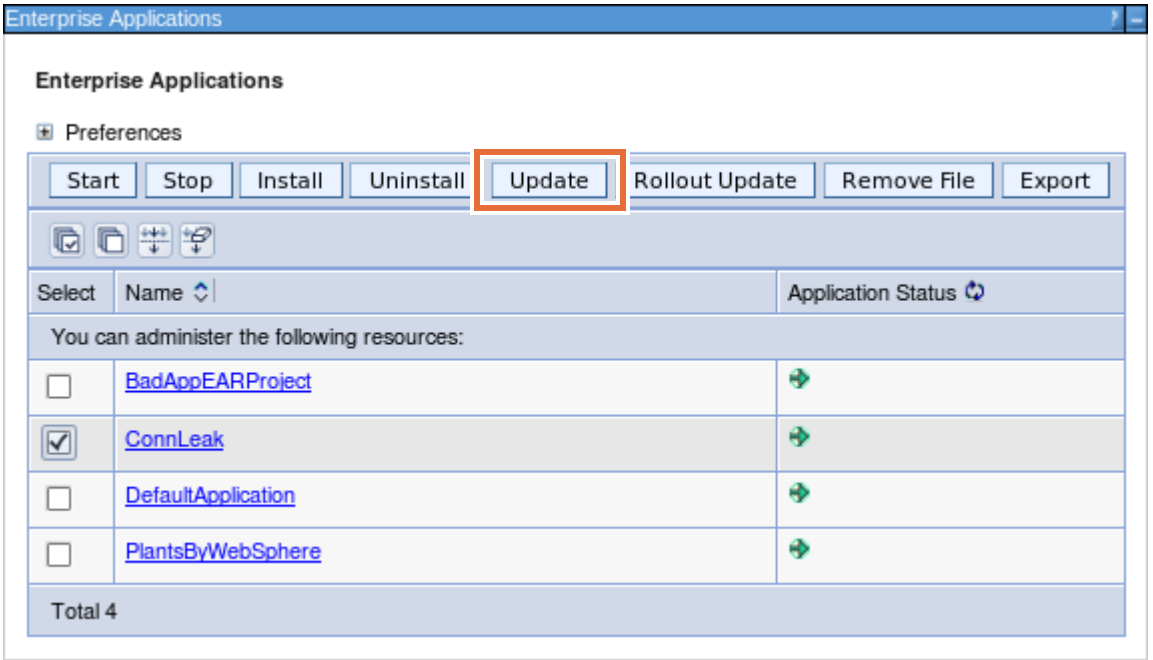
__ 3. Next, deploy the updated ConnLeak application by using the administrative console.

__ a. Log on to the administrative console, and start **server1**.

__ b. Click **Applications > Applications Types > WebSphere enterprise applications**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Check the check box next to **ConnLeak** and click **Update**.



__ d. In the Upload the replacement application box, select **Local file system** radio.

__ e. Click **Browse** and navigate to `/usr/Student/ConnLeak`. Select the `ConnLeak_solution.ear` application and click **Open**.

__ f. Click **Next** to begin the installation process.

__ g. Select the **fast path**, and click **Next**.

__ h. Click **Step 3: Summary**. You are taking all of the defaults through all of the installation steps. No changes are required.

__ i. Click **Finish**.

__ j. Wait for the application to install successfully.

__ k. **Save** the changes to the master configuration.

__ l. Wait for the node to synchronize and click **OK**.

Section 5: Testing the updated application

In this section, you run the simple load tester application against the updated ConnLeak application to verify that the connection leak problem is resolved.

__ 1. Delete the `trace.log` file for server1.

__ a. Stop **server1** from the administrative console.

__ b. Use a file browser to navigate to: `<profiles_root>/profile1/logs/server1`

__ c. Right-click `trace.log` and select **Move to Trash**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Start server1.

__ a. In the administrative console, select **Servers > Server Types > WebSphere application servers**.

__ b. Check **server1** and click **Start**.

__ c. Wait for the server to start.

__ 3. Use the LoadTester application again to generate a load against the updated ConnLeak application.

__ a. Open a Terminal window.

__ b. Change directory to: `/usr/Student/ConnLeak`

__ c. Enter the following command all on one line:

```
/opt/IBM/WebSphere/AppServer/java/bin/java -jar LoadTester.jar 50 5
http://localhost:9080/connleak/LeakServlet
```

__ d. In the Terminal window, wait for the message `press CTRL-C to abort test...` before proceeding.

__ e. Press **CTRL-C**.

__ 4. Review the new trace output.

__ a. Search for the string `Connection Leak` in the trace output. You no longer see any Connection Leak stack traces.

Section 6: Clean up the environment

In this section, you undo the changes to the environment that were required for this exercise.

__ 1. Change the Connection timeout connection pool property for the PLANTS data source back to 180 seconds.

__ a. Using the administrative console, click **Resources > JDBC > Data sources > PLANTS > Connection Pool Properties**.

__ b. Change the Connection timeout value back to `180`.

__ c. Click **OK** and save the changes to the master configuration.

__ 2. Disable Connection Leak Logic tracing on server1.

__ a. Using the administrative console, navigate to **Troubleshooting > Logs and trace > server1 > Change log detail levels**.

__ b. Remove the `WAS.j2c=finest` string from the trace string field and leave only: `*=info`

__ c. Click **OK** and save the changes to the master configuration.

__ 3. Configure HPEL for server1.

__ a. From the administrative console, click **Troubleshooting > Logs and trace > server1**.

__ b. Click **Switch to HPEL Mode**.

__ c. Click **Save** to save directly to the master configuration.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Wait for the nodes to synchronize and click **OK**.

__ e. Restart **server1**.

__ 4. Uninstall the **ConnLeak** application.

__ a. In a Terminal window, navigate to: `<profiles_root>/profile1/bin`

__ b. Enter the following command:

```
./wsadmin.sh -f /usr/Student/ConnLeak/uninstall_ConnLeakApp.py -user
wasadmin -password web1sphere
```

__ c. Wait until you see the following message, and for the node synchronization to complete.

```
ADMA5013I: Application ConnLeak uninstalled successfully.
```

End of exercise

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

This exercise demonstrates how to resolve a common connection leak by using the diagnostic tracing that is provided with WebSphere Application Server. The trace information was used to pinpoint the application code where the leak was originating. The connection leak was resolved by changing the type of connection that the application uses from shared to unshared.

In general, shared connections are the default and are preferred over unshared connections. This exercise used shared connections and a low timeout value to generate a connection leak scenario for illustrative purposes only.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 9. Troubleshooting security problems

What this exercise is about

This exercise demonstrates how to configure administrative and application security by using an LDAP server as the user registry. You then learn how to troubleshoot some common security problems.

What you should be able to do

At the end of this exercise, you should be able to:

- Detect security problems
- Examine and analyze log files for security messages
- Configure tracing on relevant WebSphere security components
- Examine and analyze trace logs
- Identify the security problems and fix them
- Analyze an SSL handshake failure
- Use wsadmin securityoff to disable security

Introduction

In this exercise, you work with a Network Deployment cell that has WebSphere administrative and application security enabled. The user registry is a local LDAP server. WebSphere security in this type of environment is called *distributed security,* which means that security servers run on multiple application servers all sharing a user registry. On your lab computers, you have a deployment manager, a node agent, and an application server (server1). Each of these servers must have a consistent security configuration; the *file synchronization service* helps ensure that the configurations are consistent. Troubleshooting in a distributed security environment is more complex than it would be for a stand-alone server. However, this topology is a more realistic environment for production deployments.

Requirements

Any additional software or tools are provided in the `Student` directory on your lab image or they are part of the computer setup process.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: In this section, you start the IBM Tivoli Directory Server on your lab image and verify that the server can be accessed.

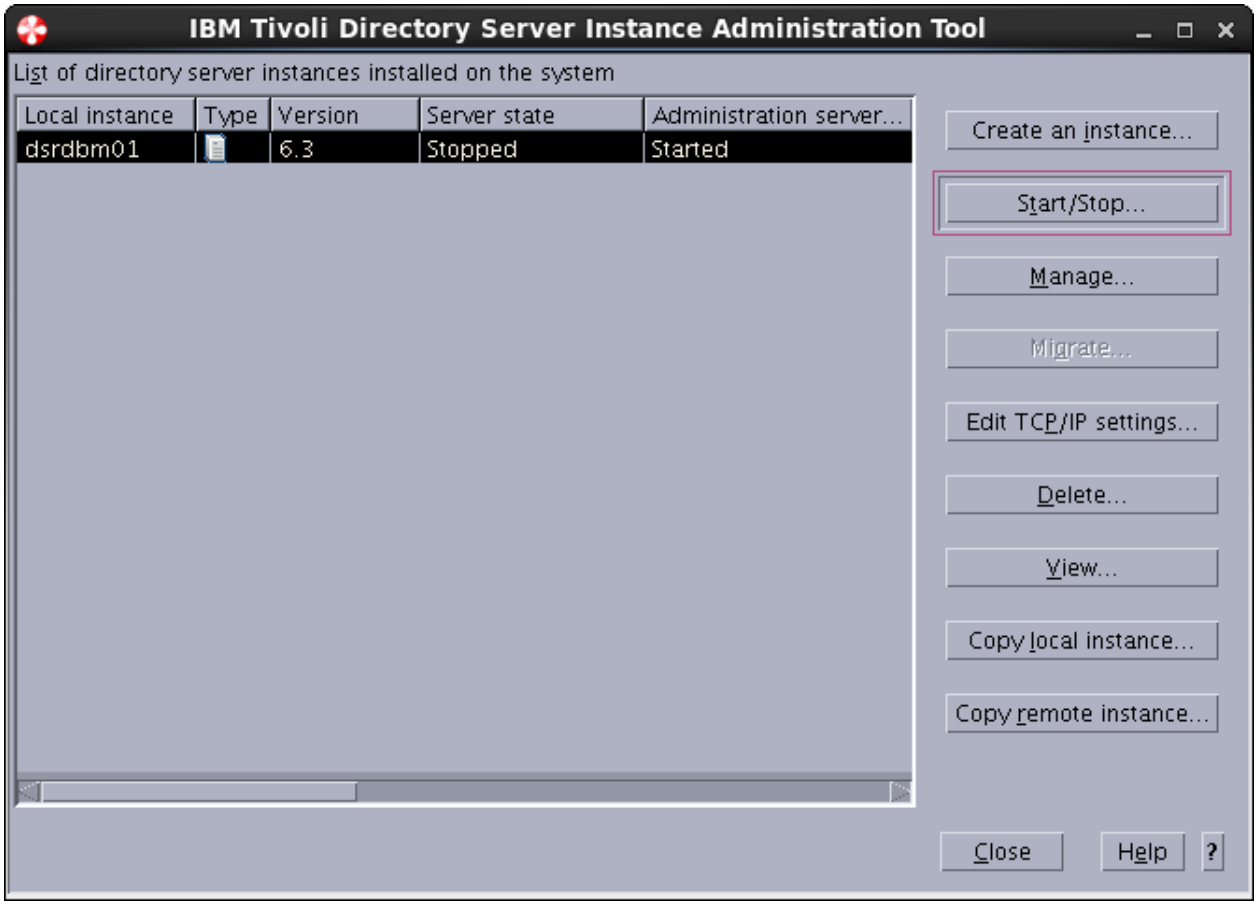
__ 1. Start the IBM Tivoli Directory Server.

__ a. From a Terminal window, navigate to the `/opt/IBM/ldap/V6.3/sbin` directory.

__ b. Enter the following command to start IBM Tivoli Directory Server Instance Administration Tool:

```
./idsxinst
```

__ c. When the administration tool window opens, click **Start/Stop**.

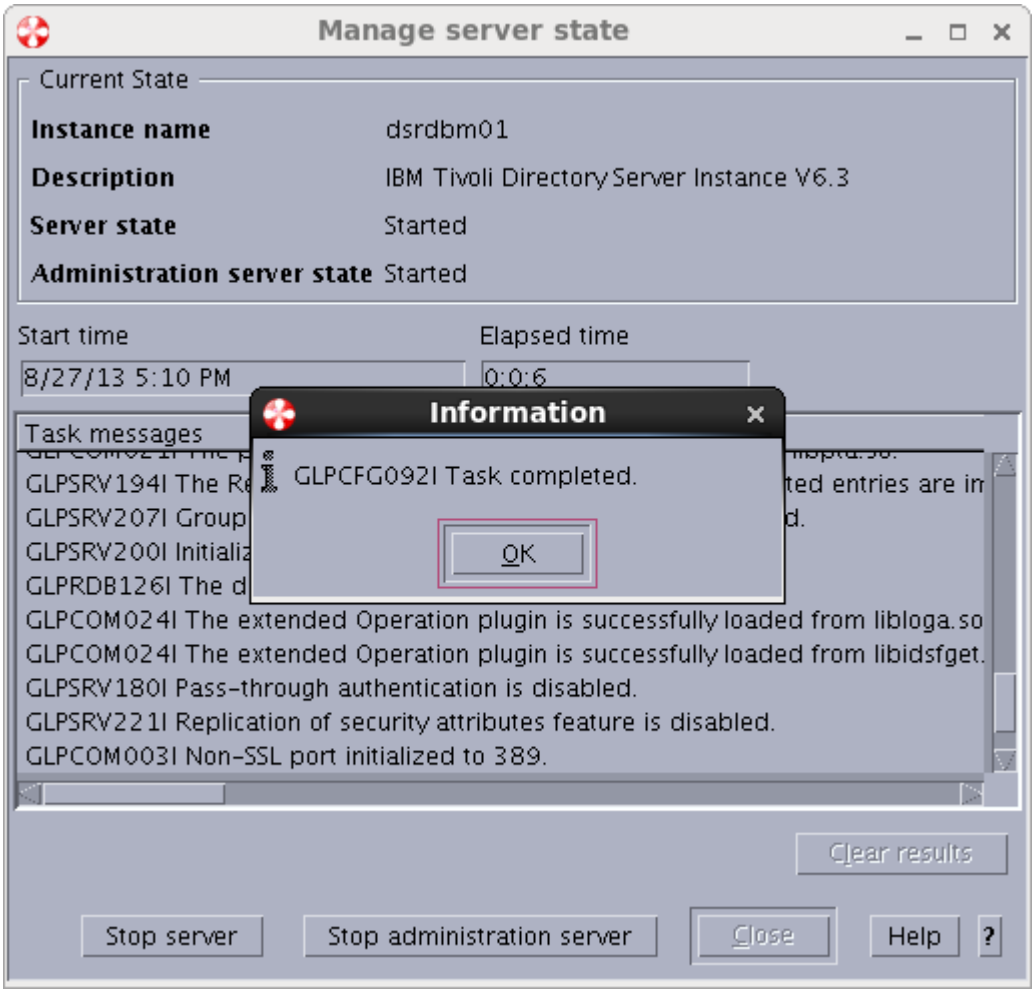


__ d. On the Manager Server State window, click **Start server**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Wait for the server state to change to **Started**.



__ f. Click **OK**.

__ g. Start the administration server if it is not already started.

__ h. Click **OK**. Click **Close** on the Manage server state window.

__ i. Do not close the IBM Tivoli Directory Server Instance Administration Tool. You are going to use it again, so minimize it.

__ 2. Use the `ldapsearch` command to verify access to the LDAP server.

__ a. Open a Terminal window and change directory to:

`/opt/IBM/ldap/V6.3/bin`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the following command:

```
./ldapsearch -b "dc=ibm,dc=com" -h washost -v "uid=wasadmin"
```

```
root@washost:/opt/IBM/ldap/V6.3/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./ldapsearch -b "dc=ibm, dc=com" -h washost -v "uid=wasadmin"
ldap_init(washost, 389)
filter pattern: uid=wasadmin
returning: ALL
filter is: (uid=wasadmin)
uid=wasadmin,cn=users,dc=ibm,dc=com
objectclass=organizationalPerson
objectclass=person
objectclass=top
objectclass=inetOrgPerson
uid=wasadmin
sn=admin
givenName=was
cn=wasadmin
1 matches
[root@washost bin]#
```

Information

`ldapsearch` opens a connection to the LDAP server specified by the `-h` option, binds, and performs a search by using specified parameters. The `-v` option turns on verbose mode with diagnostics written to standard output. The `ldapsearch` command can be used as a simple tool for verifying access to a local or remote LDAP server.

In general, you would want to encrypt connections to the user registry by using SSL. In this case, you must configure `ldapsearch` to initiate an SSL session with the LDAP server. For details on how to configure SSL, read the documentation for the ldapsearch tool at:

`http://docs.sun.com/source/816-6400-10/lsearch.html#wp19890`

Section 2: Verify the startup of the deployment manager and node agent

In the following steps, you verify that both the deployment manager and the node agent in your cell can start successfully without any errors.

__ 1. Restart the deployment manager.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./stopManager.sh -user wasadmin -password web1sphere`

__ c. Wait for the deployment manager to stop.

__ d. Using a file browser, navigate to: `<profiles_root>/Dmgr/logs/dmgr`

__ e. Delete all of the log files in the `dmgr` directory.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Back in the Terminal window, enter the command: `./startManager.sh`

__ g. When the dmgr starts successfully, open the `SystemOut.log` file of the deployment manager with gedit.

__ h. Examine the `SystemOut.log` file carefully and verify that there are ***no*** error messages, exceptions, or stack traces.

__ i. Look for the security-related informational messages that confirm the initialization of security. In particular look for the following messages:

```
SecurityConne A JSAS0006I: Security connection interceptor initialized.
ServiceInit A JSAS0001I: Security configuration initialized.
ServiceInit A JSAS0003I: Authentication mechanism: LTPA
ServiceInit A JSAS0004I: Principle name: defaultWIMFileBasedRealm/
CSIClientRI A JSAS0007I: Client request interceptor registered.
CSIServerRI A JSAS0008I: Server request interceptor registered.
SecurityCompo A JSAS0009I: IOR interceptor registered.
. . .
distSecurityCI securityServiceStarted is true
distSecurityCI SECJ0243I: Security service started successfully
distSecurityCI SECJ0210I: Security enabled true
```

```
SecurityConne A   JSAS0006I: Security connection interceptor initialized.
ServiceInit   A   JSAS0001I: Security configuration initialized.
ServiceInit   A   JSAS0003I: Authentication mechanism: LTPA
ServiceInit   A   JSAS0004I: Principal name: defaultWIMFileBasedRealm/
CSIClientRI   A   JSAS0007I: Client request interceptor registered.
CSIServerRI   A   JSAS0008I: Server request interceptor registered.
SecurityCompo A   JSAS0009I: IOR interceptor registered.
...

distSecurityC I   securityServiceStarted is true
distSecurityC I   SECJ0243I: Security service started successfully
distSecurityC I   SECJ0210I: Security enabled true
```

__ j. Close the `SystemOut.log` file.

__ 2. In a similar fashion, restart the profile1 node agent.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./stopNode.sh -user wasadmin -password web1sphere`

__ c. Using a file browser, navigate to: `<profiles_root>/profile1/logs/nodeagent`

__ d. Delete all of the log files in the `nodeagent` directory.

__ e. Back in the Terminal window, enter the command: `./startNode.sh`

__ f. When the node agent starts successfully, open the `SystemOut.log` file of the node agent by using gedit.

__ g. Examine the `SystemOut.log` file carefully and verify that there are no error messages, exceptions, or stack traces. You see security-related messages similar to the ones in the deployment manager's `SystemOut.log` file.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ h. Refresh the `SystemOut.log` file, and make sure that you see messages that indicate the node agent is successfully synchronizing with the dmgr.

`NodeSyncTask A ADMS0003I: The configuration synchronization completed successfully.`

__ i. Close the `SystemOut.log` file.

__ 3. Stop server1.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./stopServer.sh server1 -user wasadmin -password web1sphere`

Section 3: Change security realm for administrative and application security

WebSphere security was previously configured on your lab image by using a stand-alone LDAP registry. However, for the first several exercises, you ran the servers with the default file-based federated repository. In this section, you change the realm to the stand-alone LDAP registry and verify that you can successfully log on to the administrative console.

__ 1. Log on to the administrative console as user ID `wasadmin` and password `web1sphere`

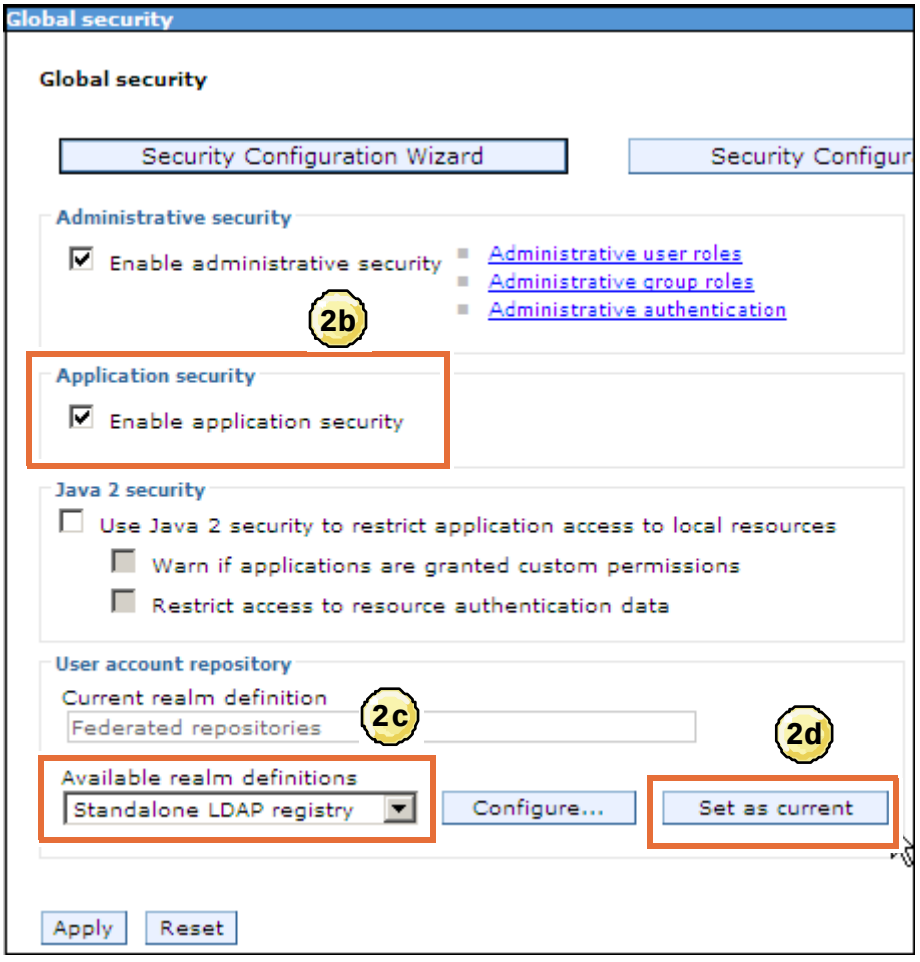
__ 2. Enable administrative and application security.

__ a. From the administrative console, select **Security > Global security**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the Global security pane, check the box for **Enable application security**.



__ c. Select **Standalone LDAP registry** from the Available realm definitions menu in the **User account repository** section.

__ d. Click **Set as current**.

__ e. Click **Apply**.

__ f. Read the messages and click the **Save** link to save the changes to the master configuration. (Make sure that you synchronize changes with the nodes.)

__ g. After the save completes and the nodes synchronize, log out of the administrative console.

Information

Console messages for a security realm change

Warning: The domain name for single signon is not defined. The web browser defaults the domain name to the host name that runs the web application. Single signon is restricted to the application server host name and does not work with other application server host names in the domain.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Warning: The security configuration is enabled or modified in a Network Deployment environment. The following steps need to be followed so that all the processes in this environment have the same security runtime settings:

1) Verify that all nodes are synchronized with these security configuration changes before stopping these processes.
2) If any node agents are currently stopped, issue a manual `syncNode` command before starting that node agent.
3) Stop all of the processes in the entire cell, including the deployment manager, node agents, and Application Servers.
4) Restart all of the processes in the cell; restart the deployment manager and node agents first, then Application Servers.

Warning: If the Restrict access to local resources option is not enabled, the Java virtual machine (JVM) system resources are not protected. For example, applications can read and write to files on file systems, listen to sockets, and exit the Application Server process. However, by enabling the Restrict access to local resources option, applications might fail to run if the required permissions are not granted to the applications.

__ 3. Restart the deployment manager.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./stopManager.sh -user wasadmin -password web1sphere`

__ c. Wait until you see the message: `ADMU4000I: Server dmgr stop completed`

__ d. As you did before, delete the deployment manager's log files from `<profiles_root>/Dmgr/logs/dmgr`.

__ e. Enter the command: `./startManager.sh`

__ f. Wait until you see the message: `Server dmgr open for e-business`

__ 4. Examine the deployment manager's `SystemOut.log` file.

__ a. Look for the security-related informational messages that confirm the initialization of security. In particular, look for the following messages:

```
SecurityConne A JSAS0006I: Security connection interceptor initialized.
ServiceInit A JSAS0001I: Security configuration initialized.
ServiceInit A JSAS0003I: Authentication mechanism: LTPA
ServiceInit A JSAS0004I: Principle name: localhost:389/
CSIClientRI A JSAS0007I: Client request interceptor registered.
CSIServerRI A JSAS0008I: Server request interceptor registered.
SecurityCompo A JSAS0009I: IOR interceptor registered.
```

```
SecurityConne A  JSAS0006I: Security connection interceptor initialized.
ServiceInit  A  JSAS0001I: Security configuration initialized.
ServiceInit  A  JSAS0003I: Authentication mechanism: LTPA
ServiceInit  A  JSAS0004I: Principal name: localhost:389/
CSIClientRI  A  JSAS0007I: Client request interceptor registered.
CSIServerRI  A  JSAS0008I: Server request interceptor registered.
SecurityCompo A  JSAS0009I: IOR interceptor registered.
```

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Notice that the **Principal name** is changed to `localhost:389`. In general, the Principal name points to the host name of the LDAP server and the LDAP port.

__ c. Scroll down and look for the following exception:

```
DefaultTokenPE HMGR0149E: An attempt to open a connection core group
DefaultCoreGroup has been rejected.
. . .
```

Notice that your IP address is going to be different.

```
DefaultTokenP E   HMGR0149E: An attempt to open a connection to core group
DefaultCoreGroup has been rejected. The sending process has a name of
was8host01Cell01\was8host01Node01\nodeagent and an IP address of /192.168.8.139.
Global security in the local process is Enabled. Global security in the sending process is
Enabled. The received token starts with ?ÓiÜo⁸N£...&§Cý¾  Òñûý°g‰Ý³. The
exception is com.ibm.websphere.security.auth.WSLoginFailedException: The user is
from a foreign realm, defaultWIMFileBasedRealm, and this foreign realm is not trusted.
Current realm is localhost:389
```

Information

The message HMGR0149E is from the high availability manager. The reason this exception is thrown is stated as “The user is from a foreign realm, defaultWIMFileBasedRealm, and this foreign realm is not trusted. Current realm is localhost:389” The realms do not match because the node agent is not fully synchronized with the deployment manager. In the next step, you restart the node agent, at which time it synchronizes with the deployment manager.

__ 5. Restart the node agent.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./stopNode.sh -user wasadmin -password web1sphere`

__ c. Wait until you see the message: `ADMU4000I: Server nodeagent stop completed.`

__ d. Using a file browser, navigate to: `<profiles_root>/profile1/logs/nodeagent`

__ e. Delete all of the log files in the `nodeagent` directory.

__ f. Back in the Terminal window, enter the command: `./startNode.sh`

__ g. Wait until you see the message: `Server nodeagent open for e-business.`

__ 6. Examine the `SystemOut.log` file of the node agent.

__ a. Use gedit to open the `SystemOut.log` file of the node agent.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Look for the following messages that indicate security configuration of the node agent initialized successfully

```
SecurityConne A JSAS0006I: Security connection interceptor initialized.
ServiceInit A JSAS0001I: Security configuration initialized.
ServiceInit A JSAS0003I: Authentication mechanism: LTPA
ServiceInit A JSAS0004I: Principle name: localhost:389/
CSIClientRI A JSAS0007I: Client request interceptor registered.
CSIServerRI A JSAS0008I: Server request interceptor registered.
SecurityCompo A JSAS0009I: IOR interceptor registered.
. . .
distSecurityCI securityServiceStarted is true
distSecurityCI SECJ0243I: Security service started successfully
distSecurityCI SECJ0210I: Security enabled true
```

```
SecurityConne A  JSAS0006I: Security connection interceptor initialized.
ServiceInit  A  JSAS0001I: Security configuration initialized.
ServiceInit  A  JSAS0003I: Authentication mechanism: LTPA
ServiceInit  A  JSAS0004I: Principal name: localhost:389/
CSIClientRI  A  JSAS0007I: Client request interceptor registered.
CSIServerRI  A  JSAS0008I: Server request interceptor registered.
SecurityCompo A  JSAS0009I: IOR interceptor registered.
...

distSecurityC I  securityServiceStarted is true
distSecurityC I  SECJ0243I: Security service started successfully
distSecurityC I  SECJ0210I: Security enabled true
```

__ c. Verify that there are no exceptions or stack traces. Notice that the HMGR exception is not thrown on restart of the node agent.

__ d. Close the `SystemOut.log` file.

Information

After security is enabled, you see the same security-related messages in the `SystemOut.log` file for the deployment manager and all node agents. However, if you are using security domains, each application server or cluster member might have a different security configuration.

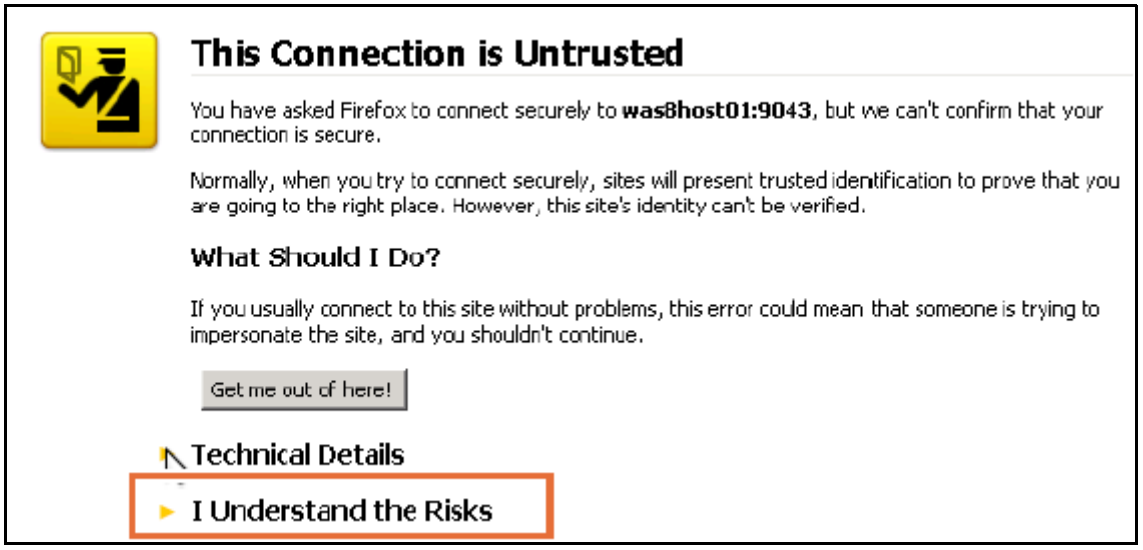
__ 7. Verify that you can log on to the administrative console.

__ a. Start a browser and enter the web address: `http://localhost:9060/ibm/console`

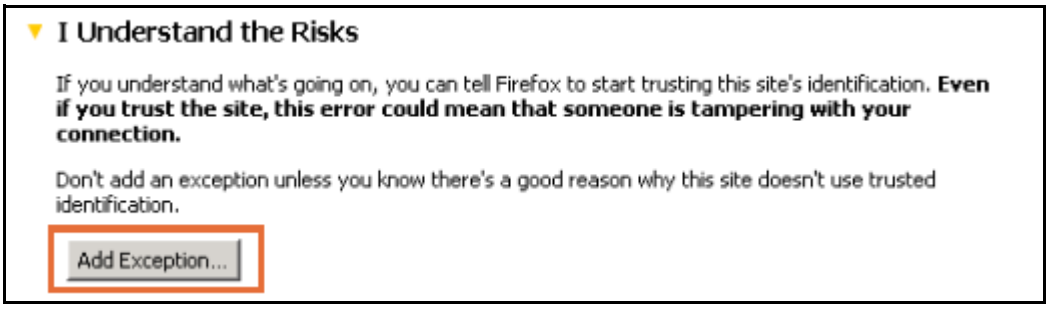
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. It is possible that the browser shows a message that the server connection is untrusted. The screen shows how Firefox shows this message: Click **I understand the Risks**.



__ c. Firefox shows this screen. It is fine to click **Add Exception...**



__ d. Log on to the administrative console with user ID `wasadmin` and a NEW password `was1edu`

Information

The user ID `wasadmin` is stored in the LDAP directory, which is now the user registry. You can verify the user ID and its password by using an LDAP browser or other LDAP administration tools.

To remind you that this authentication data is stored in a different user repository, the password is changed to `was1edu` only in the LDAP user registry.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

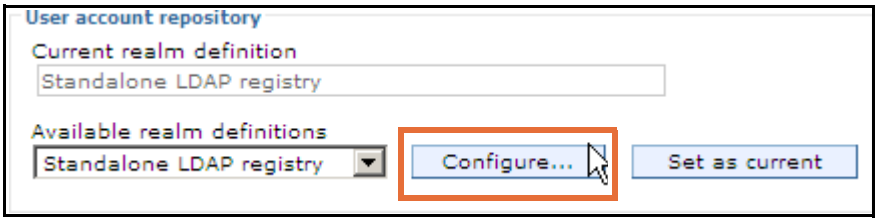
Section 4: Simulate an inaccessible LDAP server and troubleshoot the problem

In this first scenario, you explore the symptoms that result when the LDAP server becomes unavailable.

__ 1. Use the administrative console to test the connection to the LDAP server.

__ a. Click **Security > Global security**.

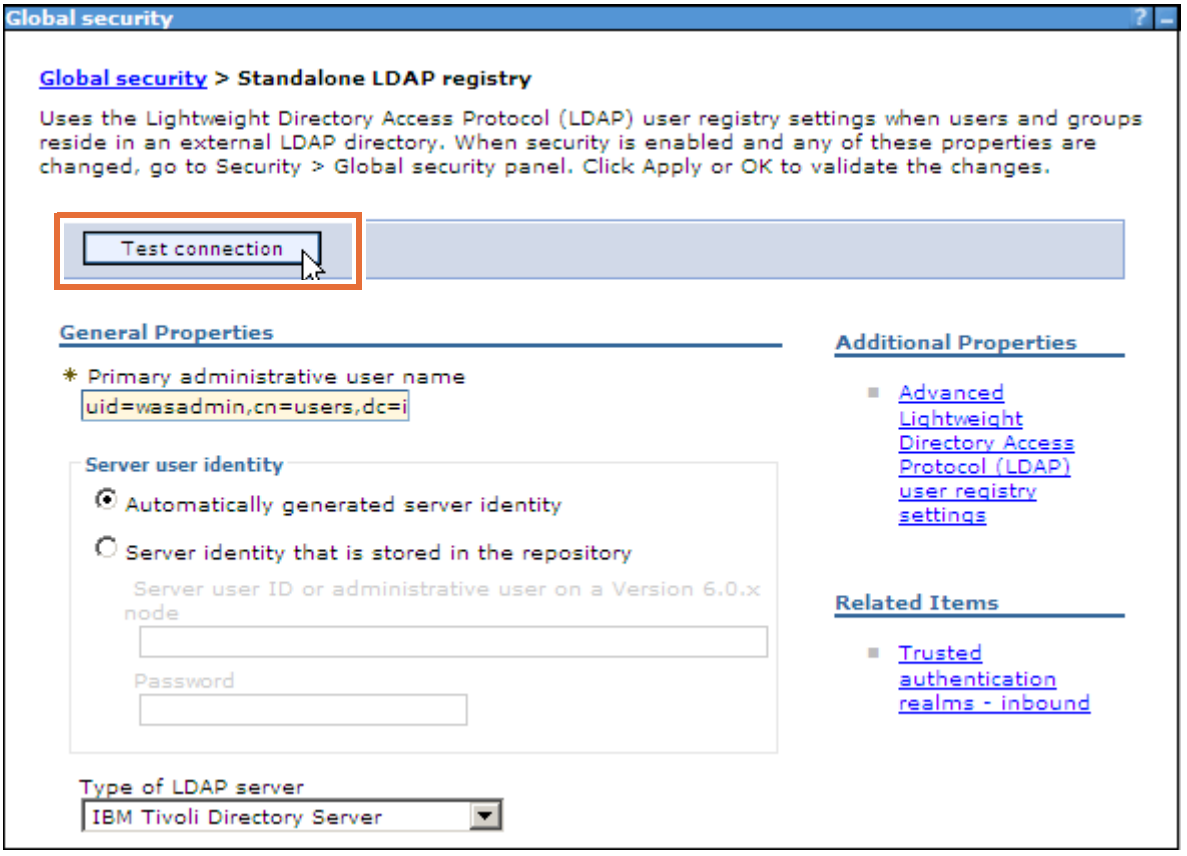
__ b. On the Global Security pane, click **Configure** beside the **Available realm definitions** box.



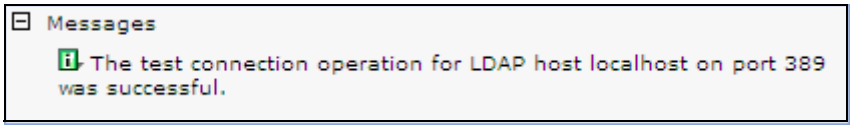
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. In the view that you now see, you can examine the configuration of the LDAP server as the stand-alone LDAP (user) registry. Click **Test connection**.



__ d. Verify that you see the message that indicates WebSphere can connect to the LDAP host.



__ e. Do not navigate away from this page. You are going to test the connection again after you stop the LDAP server.

Important

If you log out of the administrative console, you will ***not*** be able to log back in after the LDAP server is shut down.

__ 2. Stop the LDAP server.

__ a. If you did not close the IBM Tivoli Directory Server Instance Administration Tool from the previous section, go to step d. Otherwise, start the tool again.

__ b. From a Terminal window, navigate to the `/opt/IBM/ldap/V6.3/sbin` directory.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Enter the following command to start IBM Tivoli Directory Server Instance Administration Tool:

```
./idsxinst
```

__ d. When the administration tool window opens, click **Start/Stop**.

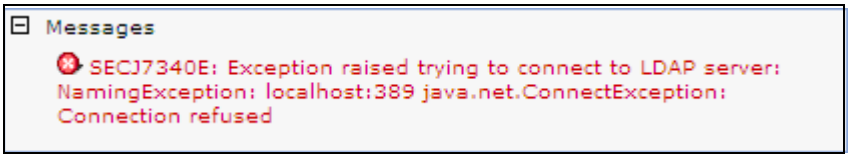
__ e. Click **Stop server**.

__ f. Do not close the IBM Tivoli Directory Server Instance Administration Tool, but minimize it.

__ 3. Test the connection.

__ a. In the administrative console, again click **Test connection**. Now you see the following exception:

```
SECJ7340E: Exception raised trying to connect to LDAP server:
NamingException: localhost:389 java.net.ConnectException: Connection
refused
```



__ b. Notice that this message can be misleading when it mentions `Naming Exception` and `Connection refused`. No suggestion that the LDAP server is down is indicated.

__ c. Do not navigate away from this page. You are going to test the connection again after you restart the LDAP server.

__ 4. Examine the messages in the `SystemErr.log` file of the deployment manager.

__ a. Using the file browser, navigate to: `<profiles_root>/Dmgr/logs/dmgr`

__ b. Open the `SystemErr.log` with gedit.

__ c. Look for messages with time stamps that coincide with the last connection failure message. You see something like the following messages:

```
SystemErr Rjavax.naming.CommunicationException:localhost:389 [Root exception is
java.net.ConnectException: Connection refused: connect]
. . .
```

```
SystemErr     R javax.naming.CommunicationException: localhost:389 [Root exception is
java.net.ConnectException: Connection refused: connect]
SystemErr     R          at com.sun.jndi.ldap.Connection.<init>(Connection.java:208)
SystemErr     R          at com.sun.jndi.ldap.LdapClient.<init>(LdapClient.java:130)
SystemErr     R          at com.sun.jndi.ldap.LdapClient.getInstance(LdapClient.java:1592)
...<Stack trace>...

SystemErr     R Caused by: java.net.ConnectException: Connection refused
```

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Information

You can research the exceptions that are shown in the console and in the log. You can look up the `SECJ7340E` message in the information center. The IBM Support Assistant can be used to look up the Java exceptions. The root exception that is identified in the log message is `java.net.ConnectException: Connection refused`, and a search of the Java documentation finds the following explanation: *"Signals that an error occurred while attempting to connect a socket to a remote address and port. Typically, the connection was refused remotely (For example, no process is listening on the remote address/port)."*

__ 5. Based on this diagnostic information, your next step might be to verify access to the computer that hosts the LDAP server and confirm that the LDAP server (IBM Tivoli Directory Server in this case) is running.

__ a. Try pinging the host IP address of the LDAP server.

__ b. Also, to exclude DNS problems, try pinging the host name of the LDAP server.

Note

You ***must*** be able to ping the computer that hosts the LDAP server in your lab environment. The LDAP server is running on localhost.

Information

If you cannot ping the host of the LDAP server and the server is running, begin investigating possible network issues. Check the `SystemOut.log` and `SystemErr.log` files for a `java.net.UnknownHostException`, which indicates a problem with the IP address itself, possibly a DNS error or **etc/hosts** file configuration error.

If you can ping the host IP address and the host name of the LDAP server, then you might try accessing the LDAP server from another LDAP client on the deployment manager computer. From a Terminal window, navigate to:

```
/opt/IBM/ldap/V6.3/bin
```

__ c. Enter the following command:

```
./ldapsearch -b "dc=ibm,dc=com" -h washost -v "uid=wasadmin
```


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. This command tries to connect to an LDAP server that is running on the host that is specified in the `-h` option.

```
root@washost:/opt/IBM/ldap/V6.3/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./ldapsearch -b "dc=ibm, dc=com" -h washost -v "uid=wasadmin"
ldap_init(washost, 389)
ldap_simple_bind: Can't contact LDAP server
[root@washost bin]#
```

This result suggests that the LDAP server is not running on the host.

__ e. Finally, if you can remotely access the LDAP server host by using a Telnet session for example; or you have local access to the LDAP server, you can issue the `netstat -anp` command. Search for the LDAP port 389. In a Terminal window, type the command:

`netstat -anp | grep ibmslapd`

```
root@washost:/opt/IBM/ldap/V6.3/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# netstat -anp | grep ibmslapd
tcp        0      0 127.0.0.1:55094             127.0.0.1:52001      ESTABLISHED 11801/ibmslapd

tcp        0      0 127.0.0.1:52001             127.0.0.1:55094      ESTABLISHED 11801/ibmslapd

tcp        0      0 :::389                      :::*                 LISTEN      11801/ibmslapd

[root@washost bin]#
```

__ f. Since the actual problem you are troubleshooting results from the LDAP server not running, you do ***not*** see the LDAP port 389. In a secure environment, it is likely that LDAP is using SSL encryption. Look for the default secure LDAP port 636.

__ 6. Resolve the problem.

__ a. Start the LDAP server. From the IDS Server Instance Administration Tool click **Start/Stop**. Then, click **Start server**.

__ b. In a terminal window, type the command: `netstat -anp | grep ibmslapd`

You now see the `ibmslapd` process is listening on port 389 as shown in the previous screen capture.

__ c. In the administrative console, click **Test connection**. Verify that the message indicates a successful connection to the LDAP server.

__ d. Log out of the administrative console.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 5: Troubleshooting authentication failures

In this section, you diagnose different authentication failures.

__ 1. Stop the deployment manager and delete all the logs in the `dmgr` directory.

__ a. In a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./stopManager.sh -user wasadmin -password was1edu`

__ c. When the deployment manager stops, navigate to: `<profiles_root>/Dmgr/logs/dmgr` and delete all the log files.

__ d. Enter the command: `./startManager.sh`

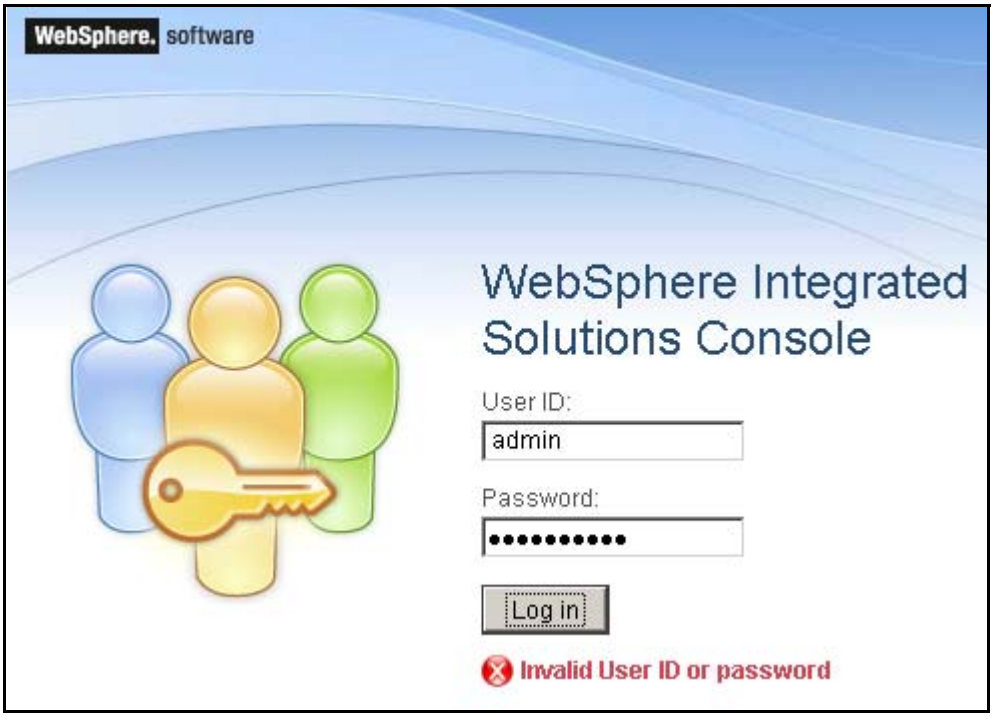
__ 2. Log on to the administrative console with an incorrect user ID.

__ a. Start a browser and enter the web address: `http://localhost:9060/ibm/console`

__ b. It is possible that the browser shows a message that the server connection is untrusted. Click **I understand the Risks**.

__ c. On the next screen, click **Add Exception...**

__ d. Log on to the administrative console with user ID: `admin` and password: `password`



__ e. Notice that the login fails, obviously because both the user ID and password are incorrect. How do you know that the user ID is not in the LDAP directory?

__ 3. Examine the `SystemOut.log` file in the `dmgr` directory for authentication error messages.

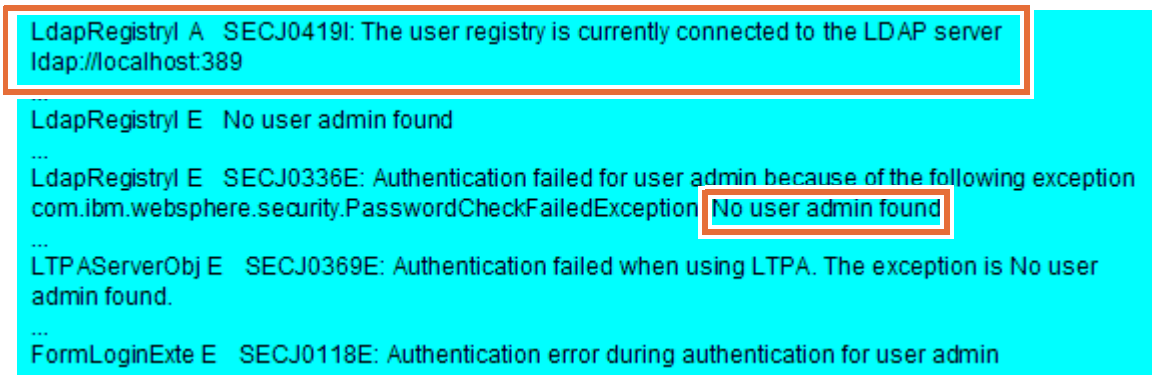
__ a. Open the `SystemOut.log` file of the deployment manager by using gedit.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Scroll down to the bottom of the log, and you see messages similar to the ones shown:

```
LdapRegistryI A SECJ0419I: The user registry is currently connected to the LDAP
server ldap://localhost:389
. . .

LdapRegistryI E SECJ0336E: Authentication failed for user admin because of the
following exception com.ibm.websphere.security.PasswordCheckFailedException: No
user admin found
. . .
```



The **SECJ0419I** message that is shown in the screen capture is important, especially in a multi-realm environment, because you need to verify which user registry is being used.

The other messages indicate that the user, admin, was not found in the registry and also that the password check failed because the user was not found.

__ c. Close the `SystemOut.log` file.

__ 4. Stop the deployment manager with an invalid user ID.

__ a. In a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Enter the command: `./stopManager.sh -user admin -password password`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/Dmgr/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./stopManager.sh -user admin -password password
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/Dmgr/logs/dmgr/stopServer.log
ADMU0128I: Starting tool with the Dmgr profile
ADMU3100I: Reading configuration for server: dmgr
ADMU0111E: Program exiting with error: javax.management.JMRuntimeException:
           ADMN0022E: Access is denied for the stop operation on Server MBean
           because of insufficient or empty credentials.
ADMU4113E: Verify that username and password information is correct.  If
           running tool from the command line, pass in the correct -username
           and -password.  Alternatively, update the <conntype>.client.props
           file.
ADMU1211I: To obtain a full trace of the failure, use the -trace option.
ADMU0211I: Error details may be seen in the file:
           /opt/IBM/WebSphere/AppServer/profiles/Dmgr/logs/dmgr/stopServer.log
[root@washost bin]#
```

__ c. Notice that the messages are helpful. In general, stop server authentication failures are resolved by:

- Verifying that the user name and password are correct
- Updating the `soap.client.props` file if you are storing user name and password there
- Verifying that the user registry can be accessed

Section 6: Troubleshooting an authorization failure

In this scenario, you create an authorization failure for a particular user by modifying the security role mappings for the default application. Then, you examine the symptoms that result.

__ 1. Delete the log files for server1.

__ a. Navigate to the directory: `<profiles_root>/profile1/logs/server1`

__ b. Delete all the log files for server1.

__ 2. For the rest of this exercise, revert to Basic mode for server1logging.

__ a. Log on to the administrative console as user `wasadmin` with password: `was1edu`

__ b. Click **Troubleshooting > Logs and trace > server1**.

__ c. Click the link, **Change log and trace mode**.

__ d. Click **Switch to Basic Mode**.

__ e. Click **Save** to save changes to the master configuration.

__ f. Wait for the nodes to synchronize and click **OK**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

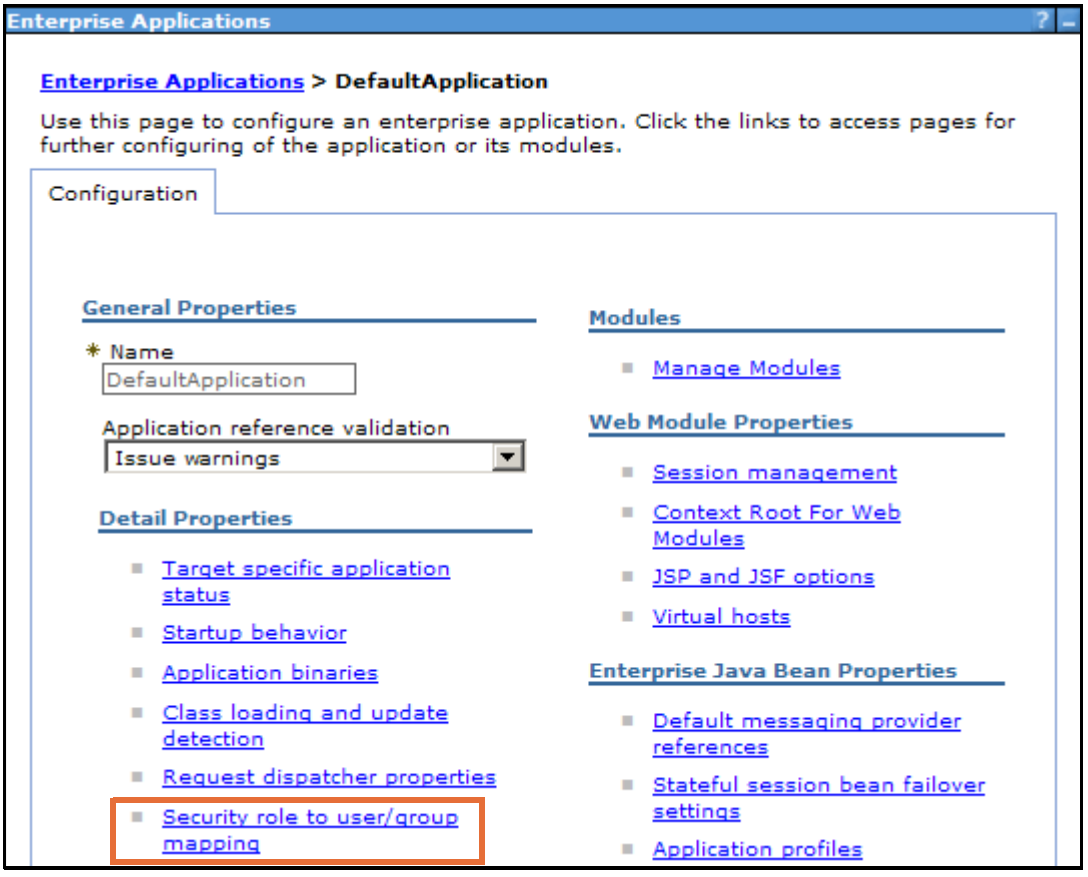
__ g. Start **server1**.

__ h. Examine the `SystemOut.log` for server1. Verify that security initialized successfully by searching for the `JSAS` and `SECJ` messages.

__ 3. Modify the security role mappings for the default application.

__ a. In the administrative console, select **Applications > Application Types > WebSphere enterprise applications** and click the link for **Default Application**.

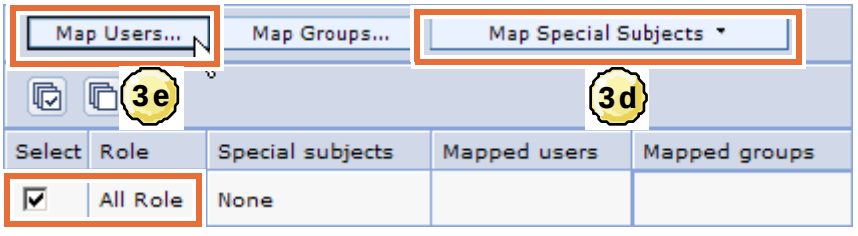
__ b. Under the **Detail Properties** section, click the link **Security role to user/group mapping**.



__ c. Select the role **All Role**.

__ d. Click **Map Special Subjects** and click **None**.

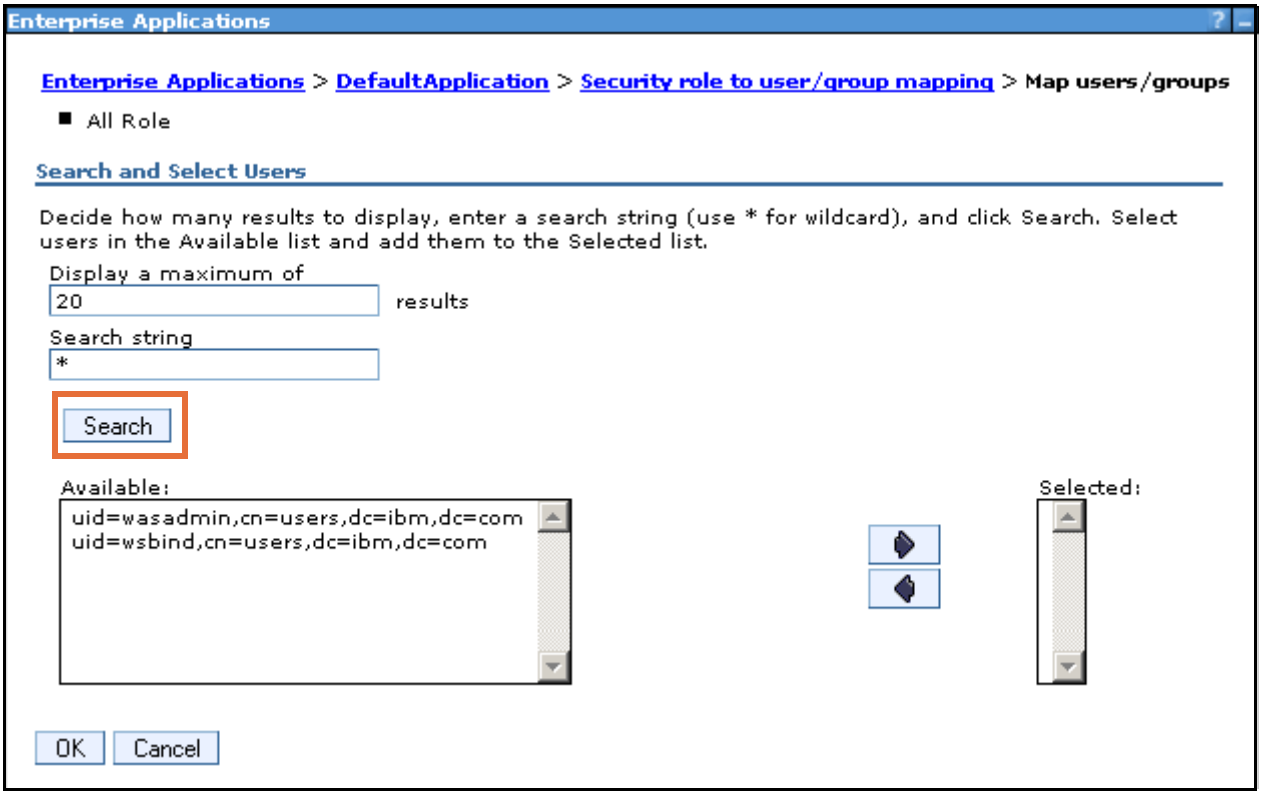
__ e. Select the role **All Role** again and click **Map Users**.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ f. Click **Search**. You see a list of user IDs in the **Available** window.

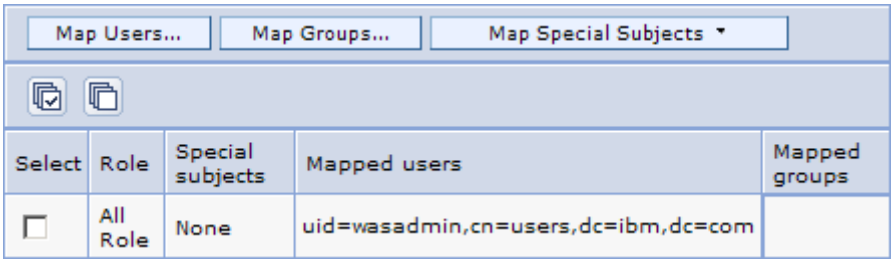


__ g. Select the **uid=wasadmin,cn=users,dc=ibm,dc=com** entry and click the arrow button:

__ h. Make sure **uid=wasadmin** shows up in the Selected list.

__ i. Click **OK**.

__ j. Your mapping must look like the following screen capture:



__ k. Click **OK** again, and click **Save** to save changes to the master configuration.

__ l. Wait for the nodes to synchronize and click **OK**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Information

You restricted access to the default application from all authenticated users to a single user.

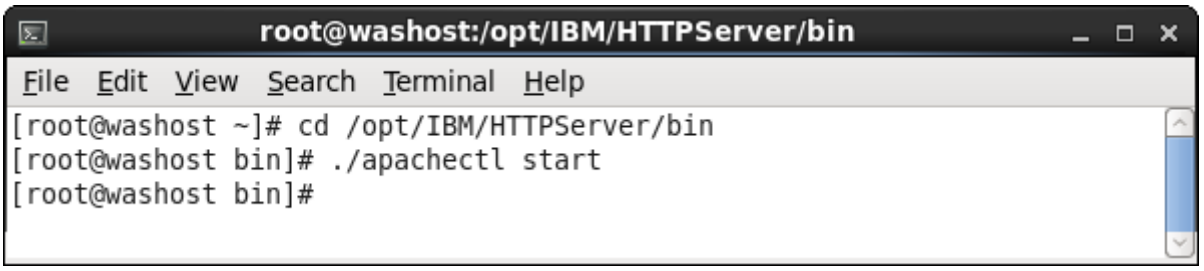
You mapped a single user, wasadmin, to the security role, All Role, which is the only role that the default application uses. Therefore, any other user from the user registry who tries to log on to the snoop servlet receives an authorization failure.

__ m. Restart **server1** from the administrative console.

__ 4. Start the IBM HTTP Server.

__ a. From a Terminal window, navigate to: `/opt/IBM/HTTPServer/bin`

__ b. Enter the command: `./apachectl start`



__ c. Enter the command to start the HTTP Administration Server:

`./adminctl start`

__ 5. Access the snoop servlet from the web browser by using port 80.

Information

With security enabled, if you try to access the snoop servlet by using the external HTTP server, you are challenged to log in. You would also be challenged if you used the web container inbound transport chain port 9080.

__ a. Open a new instance of a Firefox web browser and remove cookies by clicking **Edit > Preferences**. Select the **Privacy** tab, and click the link **clear your recent history**. Select **Everything** and click **Clear Now**.

__ b. Enter the web address: `http://localhost/snoop`

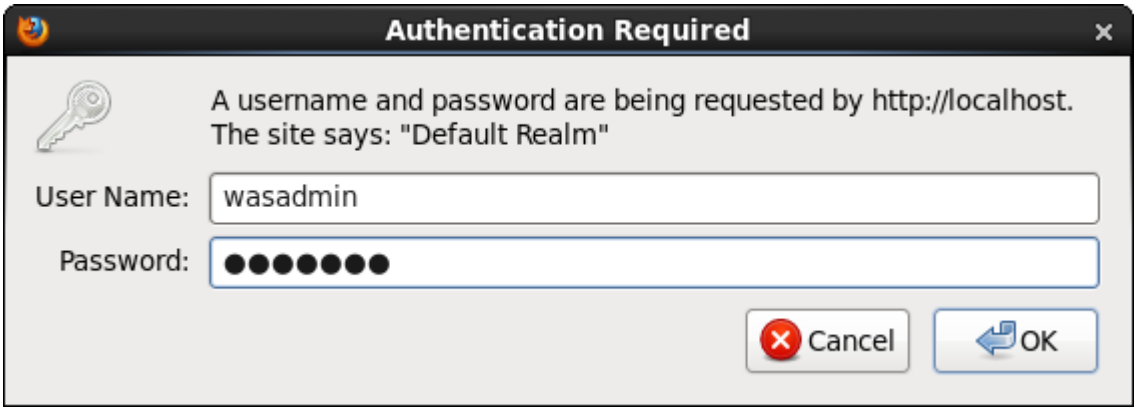
__ c. When you see the basic authentication challenge, enter the following information:

- User Name: `wasadmin`
- Password: `was1edu`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Click **OK**.



__ d. This authentication succeeds because you mapped the user wasadmin to the security role for this application. Close the browser.

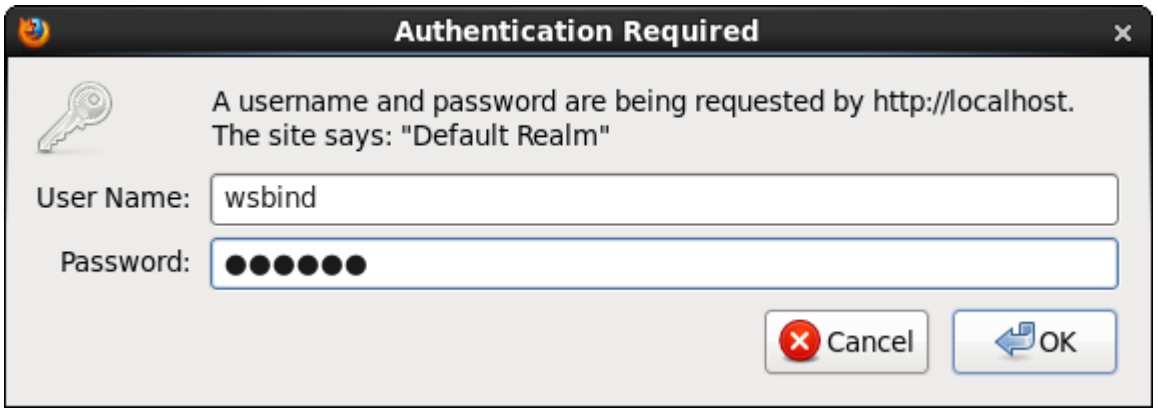
__ e. Next, try logging in as the other user in the LDAP user registry, wsbind. Open a new Firefox web browser and remove cookies by clicking **Edit > Preferences**. Select the **Privacy** tab, and click the link **clear your recent history**. Select **Everything** and click **Clear Now**.

__ f. Enter the web address: `http://localhost/snoop`

__ g. When you see the basic authentication challenge, enter the following information:

- User Name: `wsbind`
- Password: `wsbind`

Click **OK**.



__ h. Now you receive `HTTP 403 Forbidden` (or `Error 403 Authorization Failed`) error message in the web browser.

__ i. Close the browser.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ j. Open the `SystemOut.log` file for server1. Scroll to the end of the file, and you see the following message:

```
WebCollaborat A SECJ0129E: Authorization failed for user wsbind:localhost:389
while invoking GET on default_host://snoop, Authorization failed,
Not granted any of the required roles: All Role
```

WebCollaborat A SECJ0129E: Authorization failed for user **wsbind:localhost:389** while invoking GET on default_host://snoop, Authorization failed,
Not granted any of the required roles: All Role

Information

It is important to notice that this message reports an **authorization** failure, not an authentication failure. The user name and password are correct, but this user does not have permission to invoke the GET method on the snoop servlet.

You can expect this failure since you modified the role mapping to create exactly this type of authorization failure. Notice that the exception comes from the Web Collaborator and not the LdapRegistry that it would if there were a problem with the user name or password. It is an instructive exercise to trace the Web Collaborator (`com.ibm.ws.security.web.WebCollaborator`) for server1 to see exactly what steps are involved in an authorization flow.

__ k. If the user name and password were not in the user registry, you would see an *authentication* error rather than an authorization error in the log file. Also, the behavior of the login prompt would be different. Open a new browser, and try logging in with an incorrect user name to see the difference.

Section 7: Tracing security components

In this section, you configure tracing on server1 to trace the Web Collaborator. You then reproduce the authorization failure from the last section, and examine the trace output.

__ 1. Configure tracing on the running instance of server1.

__ a. In the administrative console, select **Troubleshooting > Logs and trace**.

__ b. Select **server1 > Change log detail levels**.

__ c. Select the **Runtime** tab. Expand **Components and Groups** and then **All Components**, scroll down to the component **com.ibm.ws**, and expand it.

__ d. Under **com.ibm.ws**, scroll down to the component **com.ibm.ws.security** and expand it.

__ e. Under **com.ibm.ws.security**, scroll down to the component **com.ibm.ws.security.web** and expand it.

__ f. Select **com.ibm.ws.security.web.WebCollaborator**.

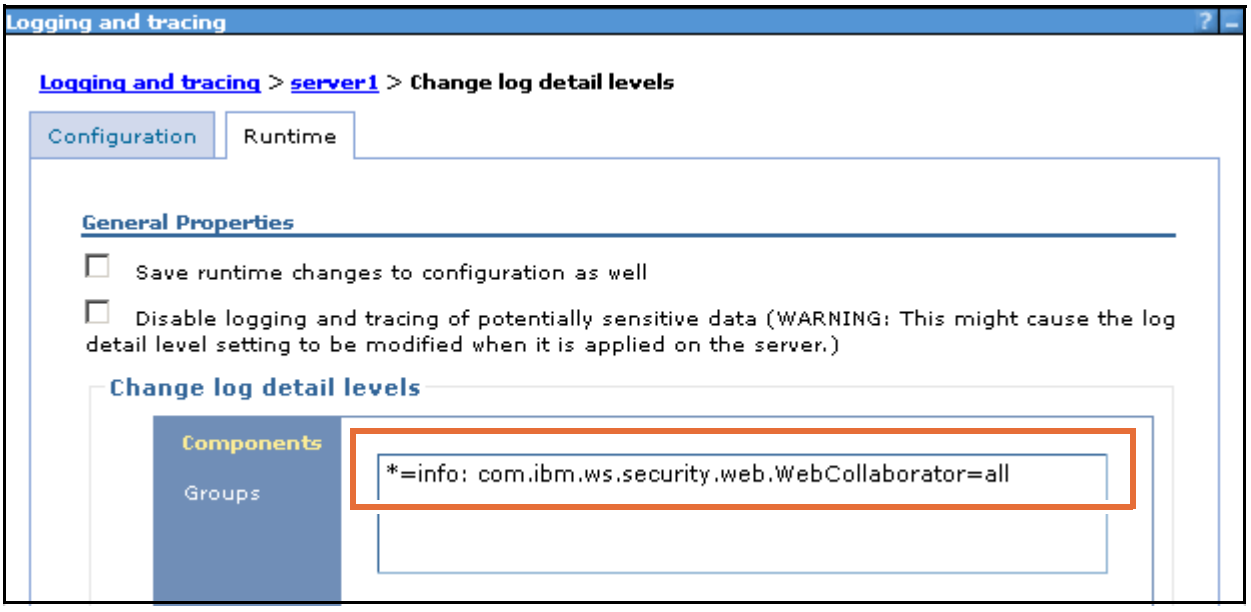
__ g. When you click the **WebCollaborator** component, a box opens with different detail levels. Click the highest level of detail, **All Messages and Traces**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

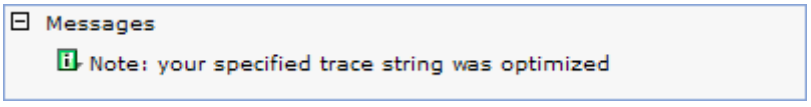
__ h. Scroll up and confirm that the trace specification has the following message under **Change log level details**:

`*=info: com.ibm.ws.security.web.WebCollaborator=all`



__ i. Click **OK**. You see the following message:

`Note: your specified trace string was optimized`



__ j. Open the `SystemOut.log` file of server1 and verify that you see a message that confirms your change to the trace specification.

```
ManagerAdmin I   TRAS0018I: The trace state has changed.
The new trace state is *=info:com.ibm.ws.security.web.WebCollaborator=all.
```

__ 2. Reproduce the snoop servlet authorization failure.

__ a. Start a new Firefox browse, clear the cookies, and type `http://localhost/snoop` in the address bar.

__ b. When you are prompted to log in, enter `wsbind` for both the user name and password and click **OK**. Verify that you see the HTTP 403 error in the browser.

__ 3. Examine the `trace.log` file of server1.

__ a. Use gedit to open the `trace.log` file of server1. By default, it is in *<profiles_root>*/profile1/logs/server1.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Scroll down to the last `SECJ0129E` message. You see something like the following messages:

```
WebCollaborat 3 URI - /snoop.GET is protected
WebCollaborat 3 Saving previous subject Subject:
. . .

WebCollaborat 3 URI checkAuthorization() failed
. . .

. . . Authorization failed for user wsbind:localhost:389 while invoking GET on
default_host://snoop . . . Not granted any of the required roles: All Role
. . .
```

```
WebCollaborat 3   URI -  /snoop.GET is protected
WebCollaborat 3   Saving previous subject Subject:
            Principal: localhost:389/wsbind
            Public Credential: com.ibm.ws.security.auth.WSCredentialImpl@1f1be74
            ...
WebCollaborat >  checkAuthorization Entry
WebCollaborat >  getAuthorizer Entry
WebCollaborat <  getAuthorizer (default) Exit
WebCollaborat <  checkAuthorization: throw AccessException Exit
WebCollaborat 3   checkAuthorization() failed, here is the message in the exception:
Authorization failed,
Not granted any of the required roles: All Role
WebCollaborat A   SECJ0129E: Authorization failed for user wsbind:localhost:389 while
invoking GET
on default_host://snoop, Authorization failed, Not granted any of the required roles: All
Role
WebCollaborat 3   Authorization check for uri: /snoop failed.
WebCollaborat <  createReply Exit
                    com.ibm.ws.security.web.DenyReply@4d01ad
[3/31/12 11:34:52:405 EDT] 0000003a WebCollaborat <  authorize Exit
                    com.ibm.ws.security.web.DenyReply@4d01ad
```

__ c. Notice that from these trace details of the Web Collaborator you can follow the complete flow of the authorization process that leads up to the authorization exception that you saw originally in the `SystemOut.log` file.

__ d. Close the `trace.log` file.

Section 8: Re-enable the file-based federated repository

In this section you switch back to the file-based federated repository as the user registry, enable administrative security, and disable application security and Java 2 security.

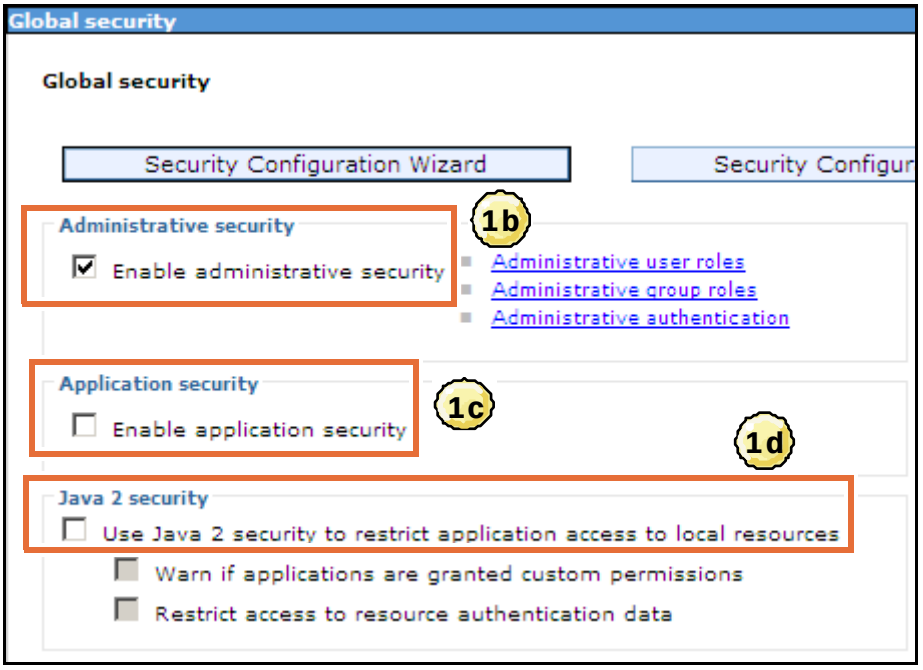
__ 1. Enable administrative security with the file-based repository.

__ a. From the administrative console, select **Security > Global security**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. On the Global security pane, select **Enable administrative security**.



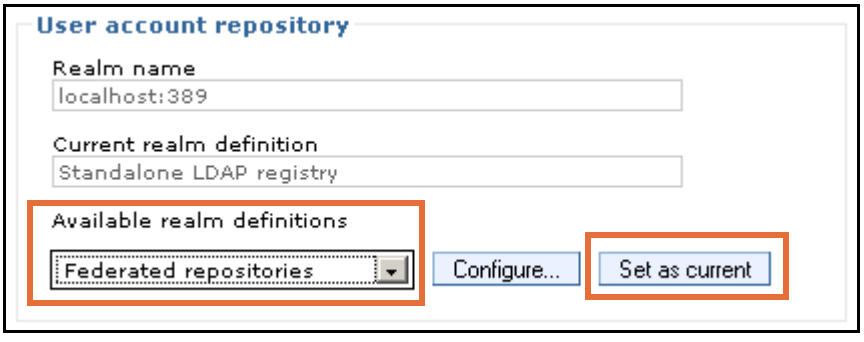
__ c. Disable (clear the check box) **Enable application security**.

__ d. Make sure that the check box **Use Java 2 security** is cleared.

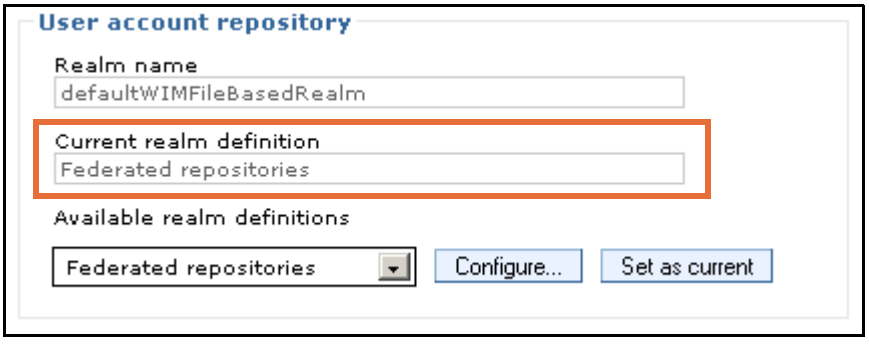
__ 2. Change the user registry back to Federated.

__ a. Scroll down on the Global security pane to the **User account repository** section.

__ b. From the Available realm definitions menu, select **Federated repositories** and click **Set as current**.



__ c. Verify that the **Current realm definition** is now **Federated repositories**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Click **Apply**.

__ e. **Save** changes to the master configuration.

__ f. Wait for the nodes to synchronize and click **OK**.

__ g. Log out of the administrative console.

__ 3. Restart the deployment manager. Notice that you must use the password `was1edu` to stop the deployment manager.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`.

__ b. Enter the command: `./stopManager.sh -user wasadmin -password was1edu`

__ c. Wait until you see the message: `Server dmgr stop completed`.

__ d. Enter the command: `./startManager.sh`

__ e. Wait until you see the message: `Server dmgr open for e-business`.

__ 4. Restart and synchronize the node agent. Notice that you must use the password `was1edu` to stop the node agent.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./stopNode.sh -user wasadmin -password was1edu`

__ c. Wait until you see the message: `Server nodeagent stop completed`.

__ d. Enter the command:

```
syncNode washost -user wasadmin -password web1sphere
```

Notice that you are authenticating to the deployment manager that is now using the file-based repository, so `web1sphere` is the required password.

__ e. Wait for the node to synchronize successfully, and enter the command: `./startNode.sh`

__ f. Wait until you see the message: `Server nodeagent open for e-business`.

__ 5. Restart server1.

__ a. Log on to the administrative console as user `wasadmin` with password `web1sphere`

__ b. Click **Servers > Server Types > WebSphere application servers**.

__ c. Since the running instance of server1 might not be aware of the new security configuration, select **server1** and click **Terminate**. Click **OK** on the confirmation dialog pane. Wait for the server to stop successfully. If the terminate step fails, you might need to end the server1 process by using: `kill -9 <PID>`

__ d. Select **server1** and click **Start**. Wait for the server to start successfully.

__ 6. Verify that the user registry is back to the file-based federated repository.

__ a. Examine the `SystemOut.log` files for the deployment manager, the node agent, and server1. Verify that the startup messages for each show as follows:

```
ServiceInit A JSAS0004I: Principal name: defaultWIMFileBasedRealm/
```

ServiceInit A JSAS0004I: Principal name: defaultWIMFileBasedRealm/

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 7. Stop the IBM Tivoli Directory Server.

__ a. From a Terminal window, navigate to the `/opt/IBM/ldap/V6.3/sbin` directory.

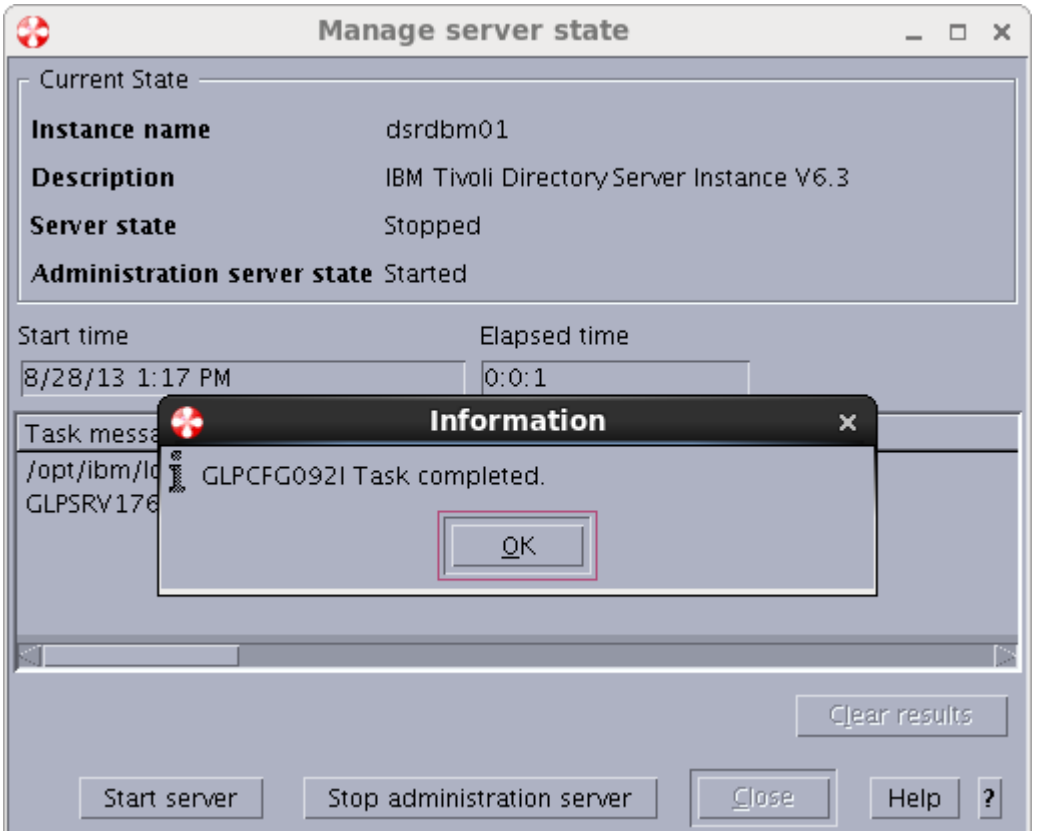
__ b. Enter the following command to start IBM Tivoli Directory Server Instance Administration Tool:

```
./idsxinst
```

__ c. When the administration tool window opens, click **Start/Stop**.

__ d. On the Manage server state screen, click **Stop server**.

__ e. Wait for the server state to change to **Stopped**, and click **OK**.



__ f. Also, stop the administration server by clicking **Stop administration server**.

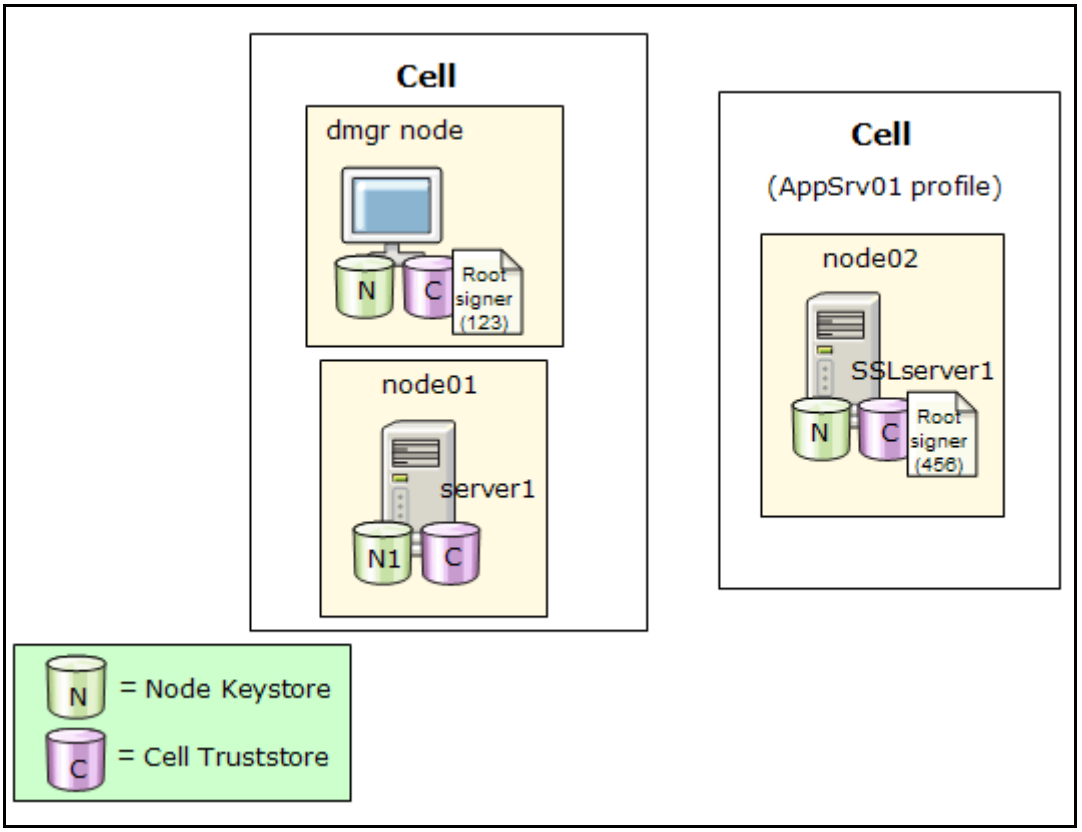
__ g. When administration server stops, click **OK**. Click **Close** on the Manage server state pane.

__ h. Close the IBM Tivoli Directory Server Instance Administration Tool by clicking **Close**, and then **Yes**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 9: Troubleshooting an SSL handshake failure

In this part of the exercise, you analyze what happens when the SSL handshake fails between two servers, and how you can diagnose the failure. As shown in this graphic, each cell has its own cell truststore. The stand-alone application server is technically considered a cell. In the cell truststore for each cell is the cell root signer certificate. As you might expect, the root signer certificates are different in each cell truststore. In this scenario, the root signer certificates from one cell are not stored in the other cell truststore. There is not a “trusted” relationship between the two cells.



You are going to use an application named **SSLTestEar** and deploy it to **server1**. For this exercise, the application is hardcoded to access the URL `https://localhost:9049`, which is the secured port on AppSrv01. You are going to create the profile AppSrv01 and configure a stand-alone server called SSLserver1. When the application uses the secured port, **SSLserver1** attempts to establish an SSL handshake. The SSL handshake fails because there is not a trusted relationship between the two cells that is established from the exchange of cell root signer certificates. The cell truststore in each cell does not have the other cell’s root signer.

__ 1. Use a script to create the AppSrv01 profile.

__ a. Start a Terminal window and change directory to: `/usr/Student/Security`

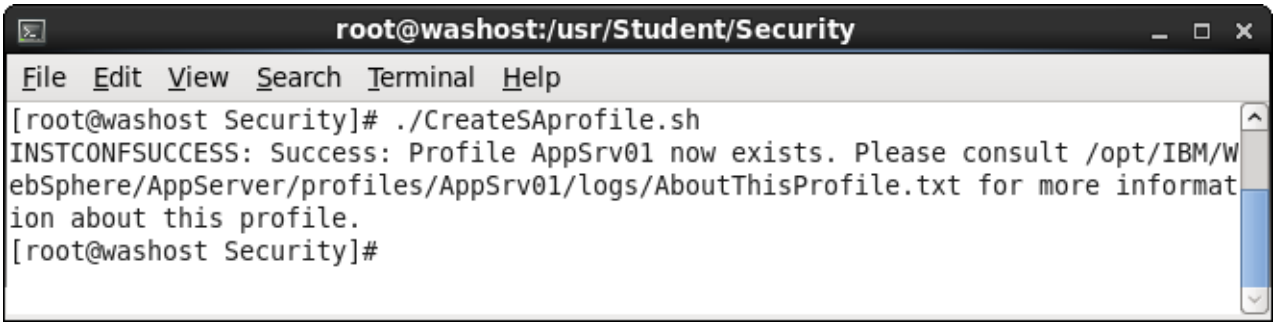
__ b. Run the command: `./CreateSAprofile.sh`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Wait until you see the following message:

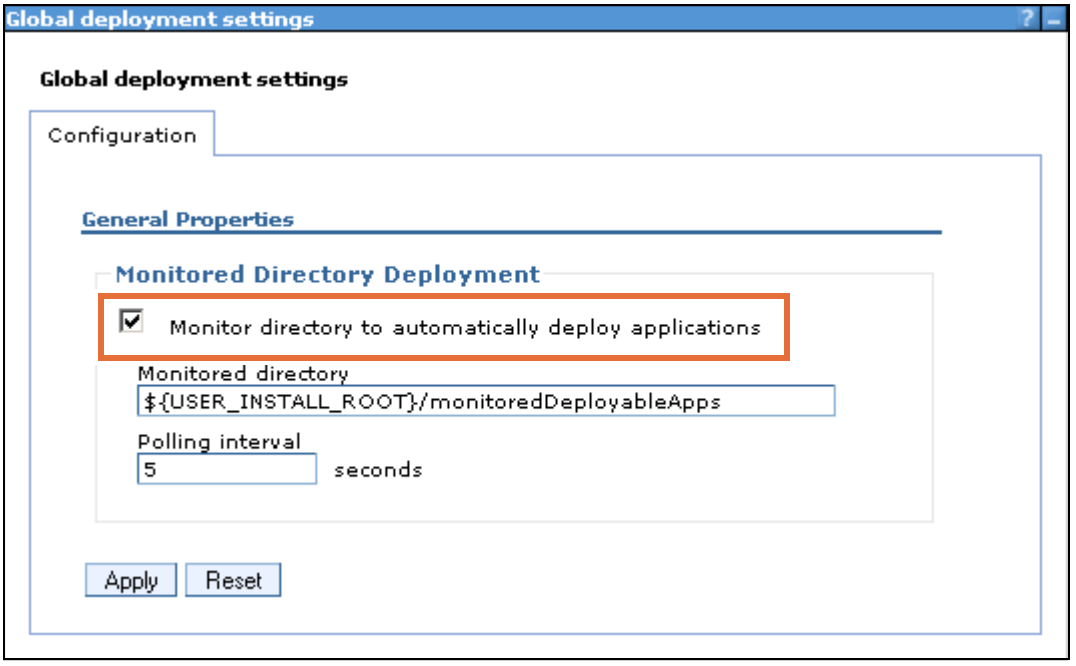
```
INSTCONFSUCCESS: Success: Profile AppSrv01 now exists.
. . .
```



__ 2. Enable monitored directory deployment.

__ a. Log on to the administrative console, and click **Applications > Global deployment settings**.

__ b. Check the box for **Monitor directory to automatically deploy applications**.



__ c. Click **Apply**.

__ d. **Save** changes to the master configuration.

__ e. Wait for the nodes to synchronize and click **OK**.

__ f. Log out of the administrative console.

__ 3. Restart the deployment manager.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./stopManager.sh -user wasadmin -password web1sphere`

__ c. Wait until you see the message: `Server dmgr stop completed`.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Enter the command: `./startManager.sh`

__ e. Wait until you see the message: `Server dmgr open for e-business`.

__ 4. Verify that the monitored directory is created, and create a directory for server1.

__ a. Use a file browser to navigate to:

`<profiles_root>/Dmgr/monitoredDeployableApps/servers`

__ b. Create a directory for **server1** by clicking **File > Create Folder**.

__ c. Name the new folder **server1**.

__ 5. Install the SSLTestEAR application by using the monitored directory.

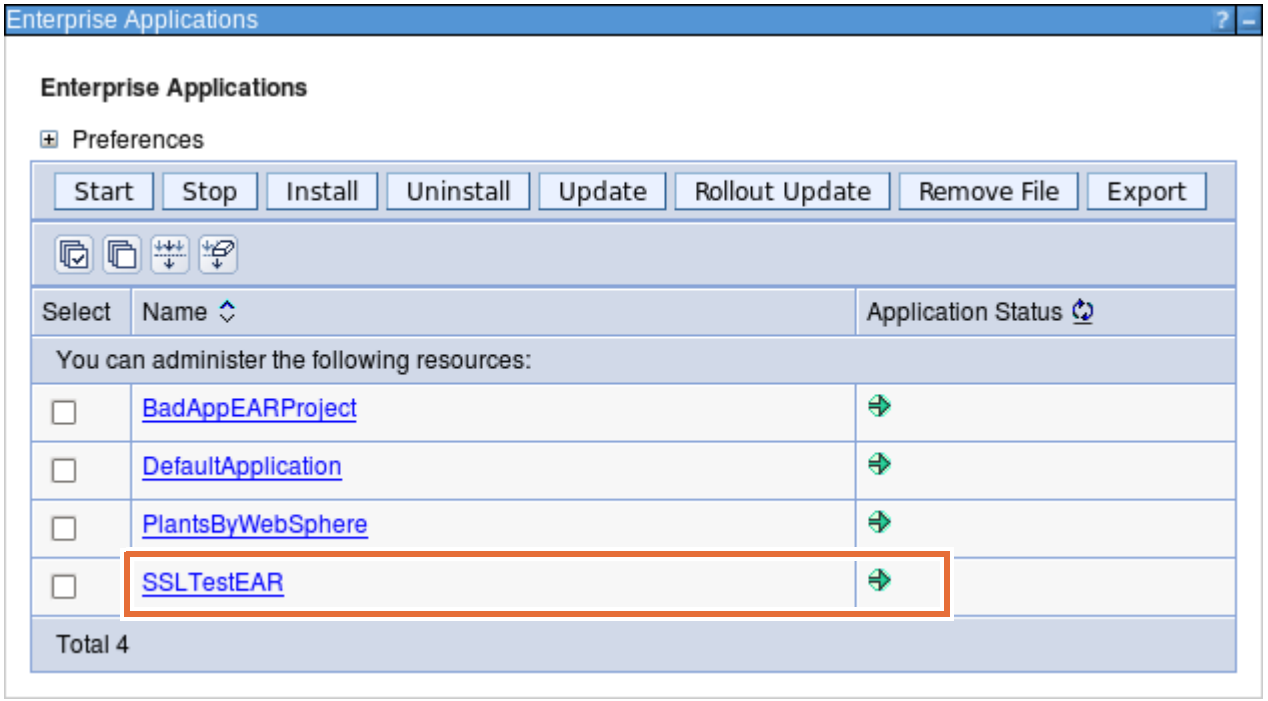
__ a. From the file browser, navigate to: `/usr/Student/Security`

__ b. Copy the `SSLTestEAR.ear` file to:

`<profiles_root>/Dmgr/monitoredDeployableApps/servers/server1`

__ 6. Verify that the application is installed and started.

__ a. Log on to the administrative console, and click **Applications > Application Types > WebSphere enterprise applications**.



__ b. Verify that the SSLTestEAR application in started. If not, select it, and click **Start**.

__ c. Log out of the administrative console.

__ 7. Now you are prepared to create an SSL handshake error. Start the stand-alone application server.

__ a. From a Terminal window, change directory to:

`<profiles_root>/AppSrv01/bin`

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ b. Enter the command: `./startServer.sh SSLserver1`

__ c. Wait for the server to start successfully.

__ 8. Log on to the administrative console for AppSvr01.

__ a. Start a new instance of a Firefox browser and enter the web address: `http://localhost:9069/ibm/console`

__ b. The browser shows a message that the server connection is untrusted. Click "**I understand the Risks**".

__ c. Click **Add Exception...**

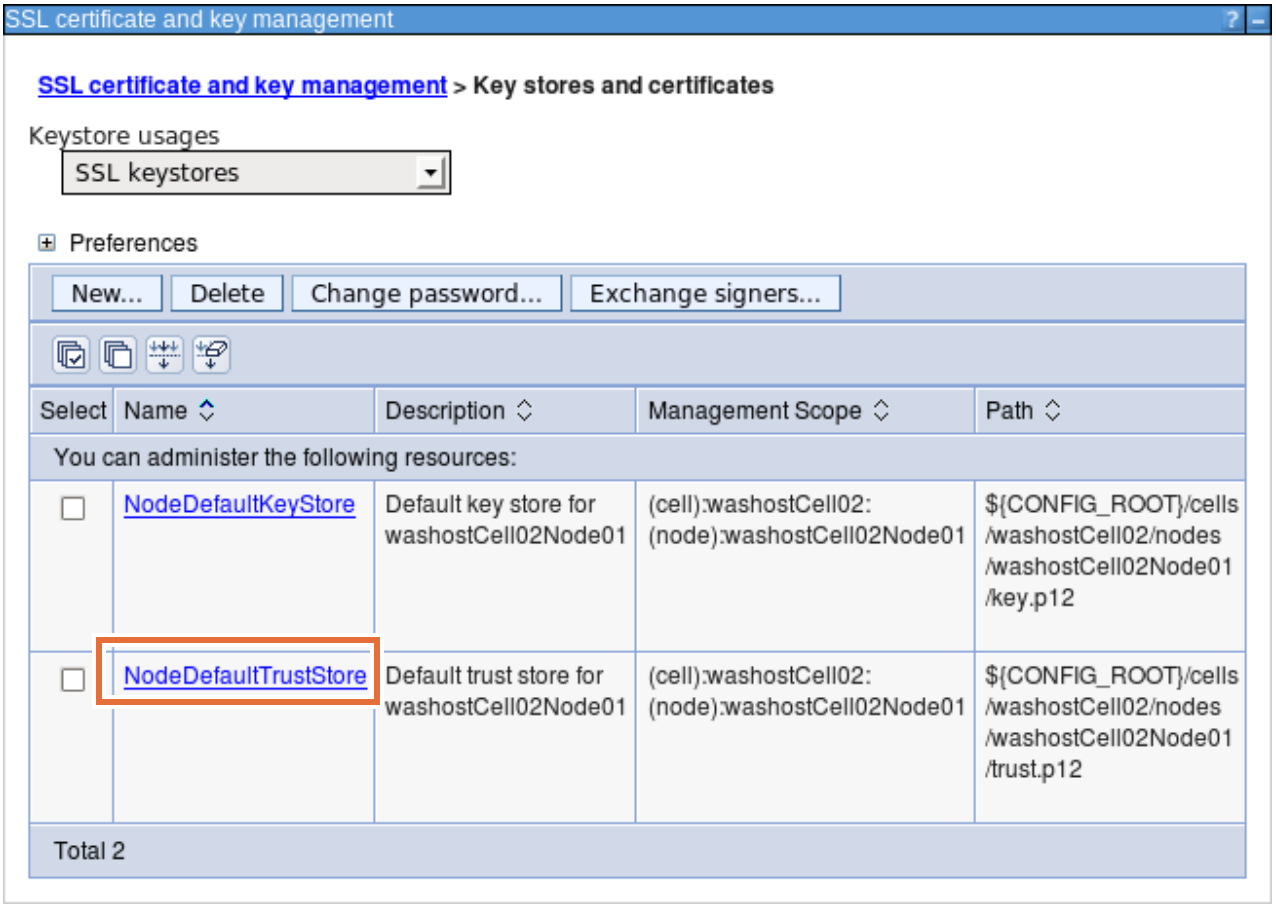
__ d. On the next screen, click **Confirm Security Exception**.

__ e. Log on as user `wasadmin` with password `web1sphere`

__ 9. Examine the signer certificates in the cell truststore of the stand-alone server.

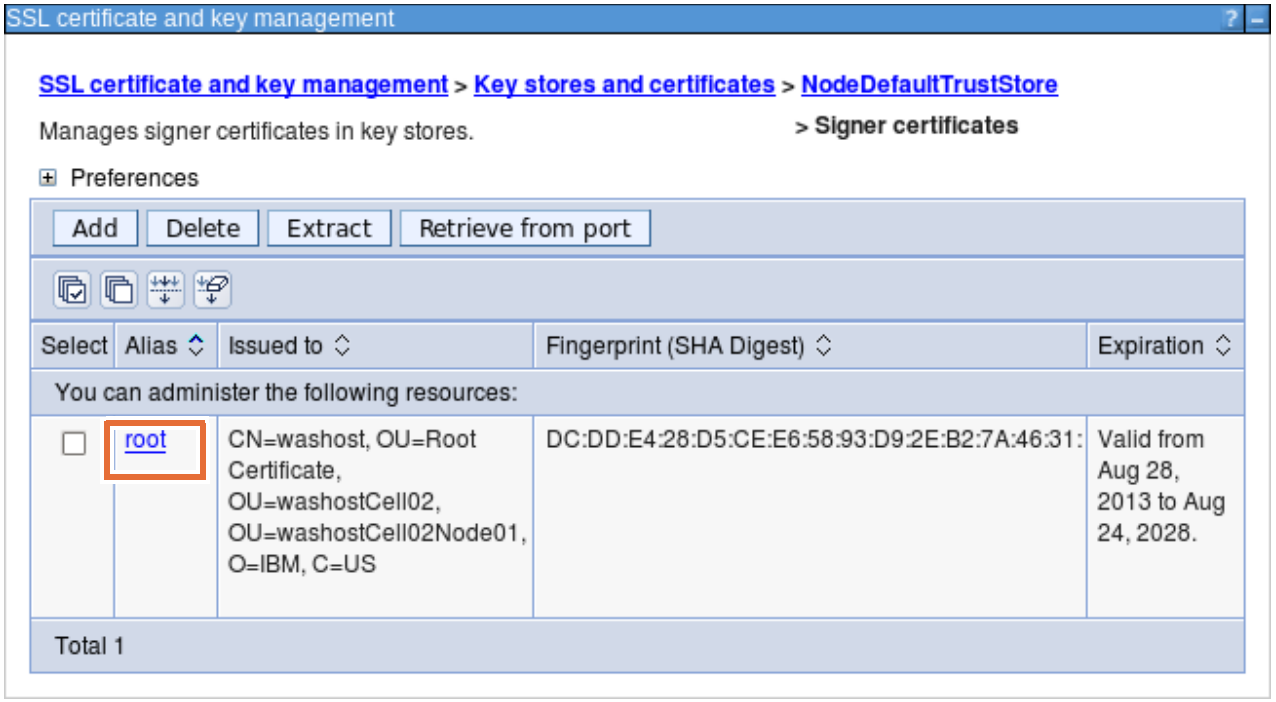
__ a. Click **Security > SSL certificate and key management**. Under **Related Items**, click **Key stores and certificates**.

__ b. Click **NodeDefaultTrustStore**. This link leads to the truststore that contains the root signer certificates.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Under Additional Properties, click **Signer certificates**. Click **root**. This link is the root signer certificate for this cell.



 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Notice that this certificate is self-signed because the **Issued to** and **Issued by** fields are the same.

__ e. Record your Serial number here: _____

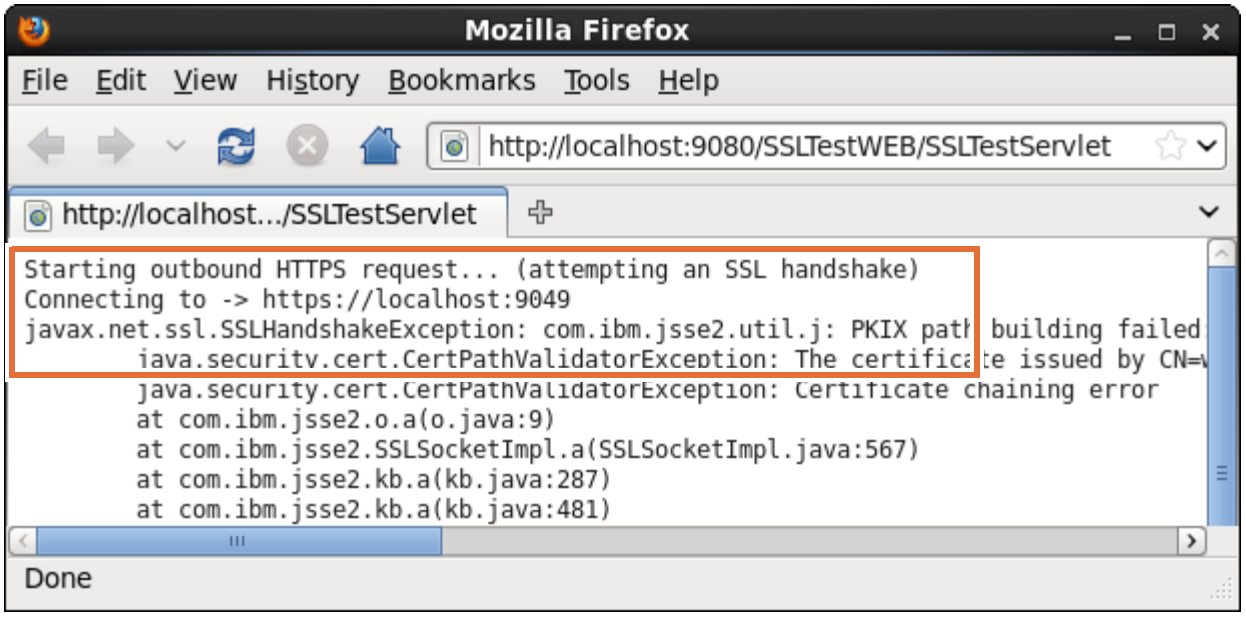
__ f. Log out of the administrative console.

__ 10. Use the SSLTestServlet to cause an SSL handshake failure.

__ a. Open a new **Firefox** web browser and remove cookies by clicking **Edit > Preferences**. Select the **Privacy** tab, and click the link **clear your recent history**. Select **Everything** and click **Clear Now**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the following web address:
`http://localhost:9080/SSLTestWEB/SSLTestServlet`



Note

If you do not see the message that is shown, restart the SSLTestEAR application.

__ c. The highlighted area shows that the request failed while starting its outbound request to the secured port (9049) on the stand-alone server. For this exercise, this application prints only some of the messages that are written to the server log. Close the web browser.

__ 11. Examine the `SystemOut.log` file for server1 on node01.

__ a. Use a file browser to navigate to: *<profiles_root>*`/profile1/logs/server1`

__ b. Open the `SystemOut.log` file by using gedit.

__ c. Search for the error code `CWPKI0022E`. You see the following message:

```
WSX509TrustMa E CWPKI0022E: SSL HANDSHAKE FAILURE: A signer with SubjectDN
"CN=washost, OU=washostCell02, OU=washostCell02Node01, O=IBM, C=US" was sent from
```


© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

```
target host:port "localhost:9049". The signer may need to be added to local trust
store . . .
```

WSX509TrustMa E CWPKI0022E: **SSL HANDSHAKE FAILURE**: A signer with SubjectDN "CN=washost, OU=washostCell02, OU=washostCell02Node01, O=IBM, C=US" was sent from target **host:port "localhost:9049".** The signer may need to be added to local trust store "/opt/IBM/WebSphere/AppServer/profiles/profile1/config/cells/washostCell/trust.p12" located in SSL configuration alias "NodeDefaultSSLSettings" loaded from SSL configuration file "security.xml". The extended error message from the SSL handshake exception is: "PKIX path building failed: java.security.cert.CertPathBuilderException: PKIXCertPathBuilderImpl could not build a valid CertPath.; **internal cause is:**
java.security.cert.CertPathValidatorException: The certificate issued by CN=washost, OU=Root Certificate, OU=washostCell02, OU=washostCell02Node01, O=IBM, C=US is not trusted; internal cause is:
java.security.cert.CertPathValidatorException: Certificate chaining error".

__ d. At the bottom of the message, notice that the certificate in question was issued from `CN=washost, OU-Root Certificate, OU=washostCell02, OU=washostCell02Node01, O=IBM, C=US`. This certificate is the root signer certificate from an “untrusted” site, `washostCell02`, specifically, the stand-alone application server, **SSLServer1**. The message also suggests that the signer certificate might need to be added to the local truststore. As of now, this root signer certificate is not in default truststore of the deployment manager cell.

Information

The SSLTestEAR application is deployed to server1 in the Network Deployment (ND) cell. The application is hardcoded to access the URL `https:localhost:9049`. Since the site (port 9049) is outside of the ND cell (the stand-alone server), it is considered untrusted. When the signer certificate was sent from the untrusted site to the ND cell, an SSL handshake attempt was made. The ND cell did not have the root signer certificate of the stand-alone server in its cell default truststore, so the SSL handshake failed.

How can this problem be solved? How can the ND cell trust the stand-alone application server? Importing the root signer certificate of the stand-alone application server to the ND cell establishes trust between the two parties. If the root signer certificate is found on both sites (the ND cell and the stand-alone server), then the trust relationship is established.

__ 12. Import the signer certificate from the stand-alone application server to the ND cell.

__ a. Start a new browser and log on to the administrative console of the deployment manager. Enter the web address: `http://localhost:9060/ibm/console`

__ a. Click **Security > SSL certificate and key management**. Under Related Items, click **Key stores and certificates**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Click **CellDefaultTrustStore**.

New... | Delete | Change password... | Exchange signers...

Select	Name	Description	Management Scope	Path
You can administer the following resources:				
☐	CMSKeyStore	CMSKeyStore for web server webserver1.	(cell):washostCell:(node):ihsnode:(server):webserver1	${CONFIG_ROOT}/cells/washostCell/nodes/ihsnode/servers/webserver1/plugin-key.kdb
☐	CellDefaultKeyStore	Default key store for washostCell	(cell):washostCell	${CONFIG_ROOT}/cells/washostCell/key.p12
☐	CellDefaultTrustStore	Default trust store for washostCell	(cell):washostCell	${CONFIG_ROOT}/cells/washostCell/trust.p12
☐	NodeDefaultKeyStore	Default key store for washostNode01	(cell):washostCell:(node):washostNode01	${CONFIG_ROOT}/cells/washostCell/nodes/washostNode01/key.p12
☐	NodeDefaultTrustStore	Default trust store for washostNode01	(cell):washostCell:(node):washostNode01	${CONFIG_ROOT}/cells/washostCell/nodes/washostNode01/trust.p12
Total 5				

__ c. Under Additional Properties, click **Signer certificates**.

__ d. Click **Retrieve from port**.

SSL certificate and key management

SSL certificate and key management > Key stores and certificates > CellDefaultTrustStore > Signer certificates

Preferences

Add | Delete | Extract | Retrieve from port

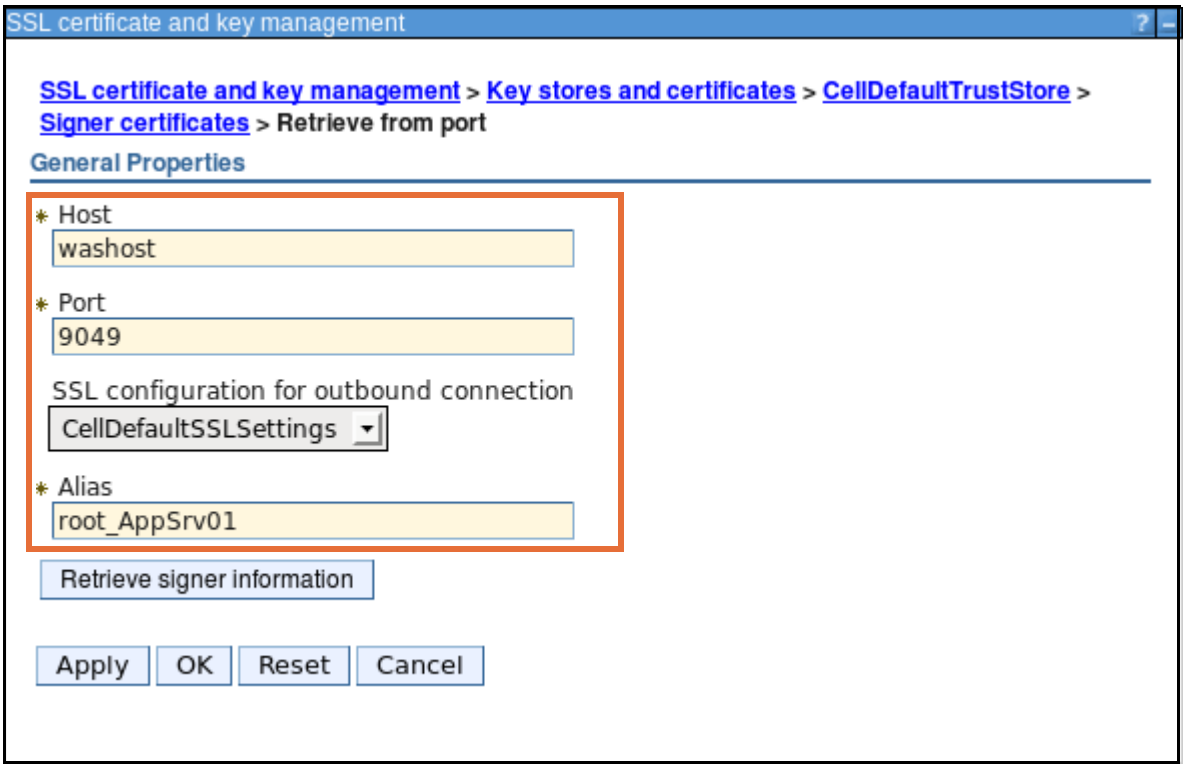
Select	Alias	Issued to	Fingerprint (SHA Digest)	Expiration
You can administer the following resources:				
☐	root	CN=washost, OU=Root Certificate, OU=washostCell, OU=washostCellManager, O=IBM, C=US	48:77:38:66:3B:89:C6:1C:01:12:30:B3:06:09:	Valid from Jun 20, 2013 to Jun 16, 2028.
Total 1				

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ e. To retrieve the root signer certificate, provide the following information:

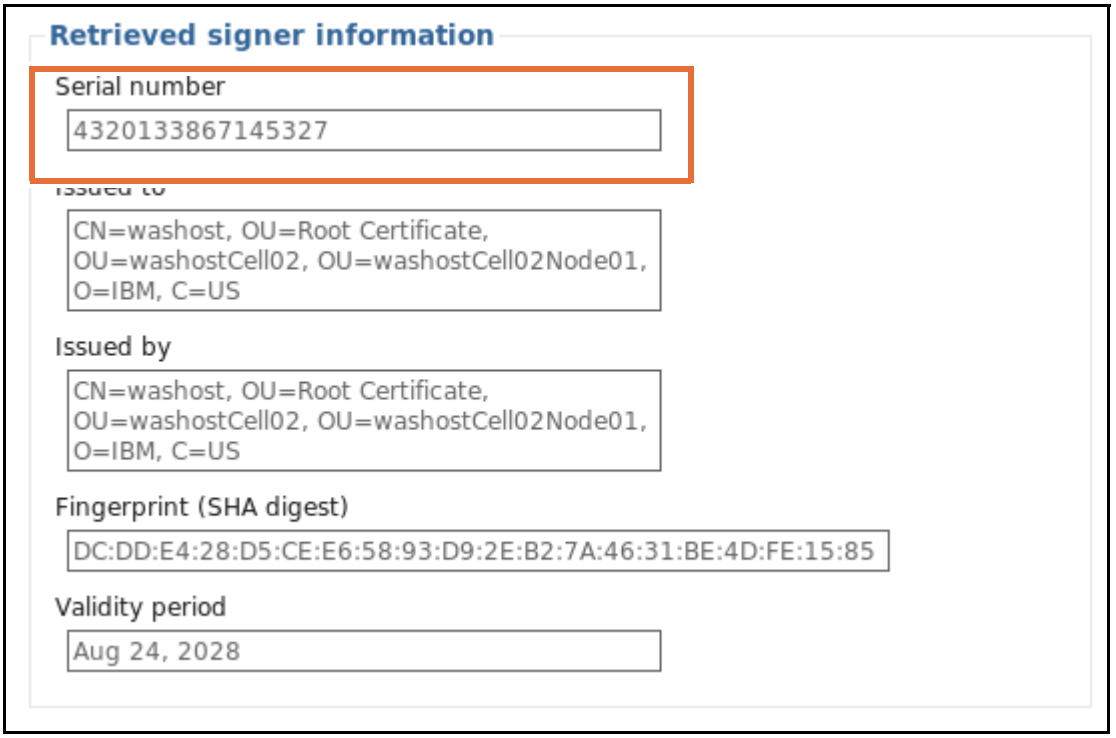
- Host: `washost`
- Port: `9049`
- Alias: `root_AppSrv01`



__ f. Click **Retrieve signer information**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. The information that is retrieved is from the root signer certificate from the stand-alone application server. You can verify that this certificate is the correct root signer certificate by comparing the serial number or the Issued by fields that you noted earlier.



Information

Notice, the certificate that was imported is ***not*** the node root signer as you might expect. It is instead the actual node certificate. Having this certificate in the truststore, though, is sufficient to validate the SSL handshake.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ h. Click **OK**. Click **Save** to persist your changes. Click **OK** again. The root signer certificate is added to the CellDefaultTrustStore.

SSL certificate and key management

SSL certificate and key management > Key stores and certificates > CellDefaultTrustStore > Signer certificates

Preferences

Add | Delete | Extract | Retrieve from port

Select	Alias	Issued to	Fingerprint (SHA Digest)	Expiration
You can administer the following resources:				
☐	root	CN=washost, OU=Root Certificate, OU=washostCell, OU=washostCellManager, O=IBM, C=US	48:77:38:66:3B:89:C6:1C:01:12:30:B3:06:09	Valid from Jun 20, 2013 to Jun 16, 2028.
☐	root_appsrv01	CN=washost, OU=Root Certificate, OU=washostCell02, OU=washostCell02Node01, O=IBM, C=US	DC:DD:E4:28:D5:CE:E6:58:93:D9:2E:B2:7A:	Valid from Aug 28, 2013 to Aug 24, 2028.
Total 2				

__ 13. Verify the trust relationship between the ND cell and the stand-alone application server.

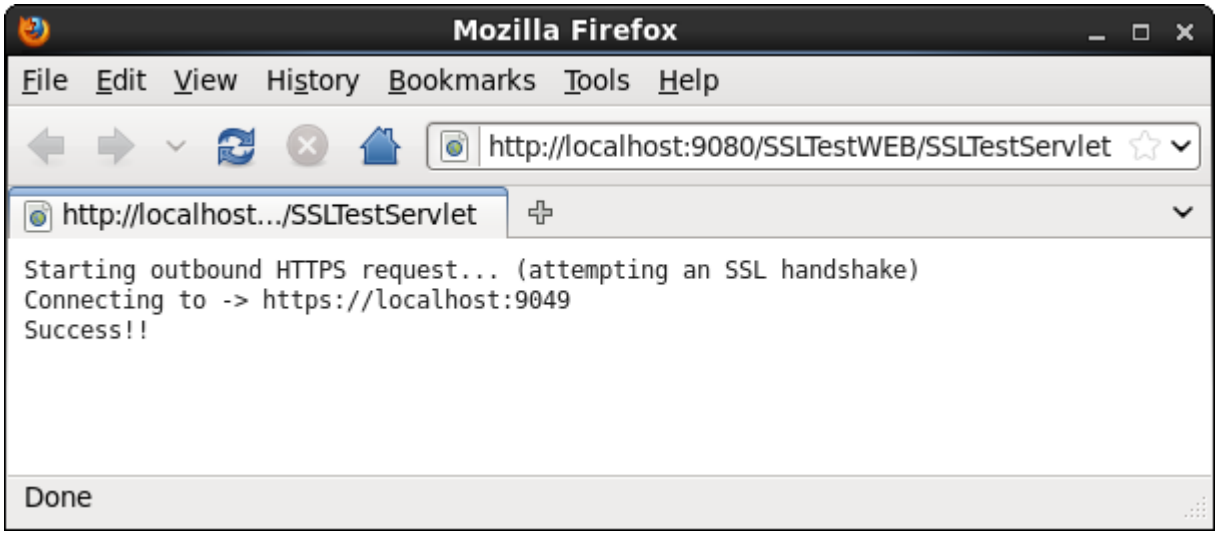
__ a. Start a new instance of **Firefox** browser and clear the cookies by clicking **Edit > Preferences**. Select the **Privacy** tab, and click the link **clear your recent history**. Select **Everything** and click **Clear Now**.

__ b. Enter the following web address:
`http://localhost:9080/SSLTestWEB/SSLTestServlet`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Because the truststore of the federated cell now has the root signer certificate from the stand-alone application server, the SSL handshake is successful. The application shows the following message:

```
Starting outbound HTTPS request... (attempting an SSL handshake)
Connecting to -> https://localhost:9049
Success!!
```



__ d. Close the web browser.

Section 10:Disable administrative security

In this section, you learn how to disable administrative security by using the wsadmin tool, but without connecting to a server. This procedure might be necessary in cases where the password of the primary administrative user is not known and there are no other administrative users. There might be cases where it is impossible to log on to the administrative console use the wsadmin tool, so you need a way to disable security to do basic administration.

You are going to use the stand-alone profile, AppSrv01, that you created in the last section to learn how to disable security.

__ 1. Disable administrative security by using the wsadmin command-line tool.

__ a. In a Terminal window, change to the directory: `<profiles_root>/AppSrv01/bin`

__ b. Enter the command: `./wsadmin.sh -conntype NONE`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. At the wsadmin prompt, type: `securityoff`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./wsadmin.sh -conntype NONE
WASX7357I: By request, this scripting client is not connected to any server process.
 Certain configuration and application operations will be available in local mode.
WASX7029I: For help, enter: "$Help help"
wsadmin>securityoff
LOCAL OS security is off now but you need to restart server1 to make it affected.


wsadmin>quit
[root@washost bin]#
```

__ d. Type: `quit`

Information

Notice the message that says "you need `to restart` server1". In the AppSrv01 profile that you created by using a script, the server name was specified as SSLserver1, not server1.

__ e. Stop **SSLserver1** by entering the command:

```
./stopServer.sh SSLserver1 -user wasadmin -password web1sphere
```

Note

The point of this scenario is that you do ***not*** know the administrative password. So in a real world situation, you would not be able to use the stop server command. Therefore, you would be required to use an operating system command to terminate the Java process of the application server.

__ f. Start SSLserver1 by entering the command: `./startServer.sh SSLserver1`

__ g. Wait for **SSLserver1** to start successfully.

__ 2. Verify that administrative security is disabled by logging on to the administrative console for profile AppSrv01 without a password.

__ a. From a browser, enter the web address: `http://localhost:9069/ibm/console`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. You notice first that there is no certificate error, and second that you are not prompted for a password. Log in as user ID: `wasadmin`

__ c. After logging on, you can verify that administrative security is cleared in the Global security panel.

__ d. You can also examine the `SystemOut.log` file for SSLserver1 to see that security is disabled. Search for the message: `SECJ0210I: Security enabled false`

__ e. Log out of the administrative console.

__ f. Stop **SSLserver1** from the command line. Notice that you can now issue the stop server command without specifying the user name and password.

__ 3. Uninstall the SSLTestEAR application by using the monitored directory.

__ a. From a file browser, navigate to: `<profiles_root>/Dmgr/monitoredDeployableApps/servers/server1`

__ b. Delete the `SSLTestEAR.ear` file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Examine the list of installed enterprise applications in the administrative console to verify that the **SSLTestEAR** application is removed.

End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In this exercise, you enabled global security by using an LDAP server as the user registry. With security enabled, you were able to explore different security-related problems that involve access to the LDAP server, authentication, and authorization. While investigating these problems, you examined and analyzed log files for security exceptions. You learned how to enable tracing on relevant WebSphere security components. In addition, you were able to examine and analyze the resulting trace logs. You learned how to identify some common security problems and how to fix them.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 10.Troubleshooting server start failures

What this exercise is about

This exercise demonstrates how to troubleshoot server start failures. You explore several scenarios where a WebSphere server fails to start or starts with errors. You troubleshoot problems that occur while starting server1, the node agent, and the deployment manager.

What you should be able to do

At the end of this exercise, you should be able to:

- Verify that an application server is failing to start
- Verify that a node agent is failing to start
- Verify that a deployment manager is failing to start
- Configure tracing on a server that fails to start
- Examine and analyze trace logs

Introduction

Sometimes servers fail to start. Possibly it is because of a bad build, or perhaps you modified your install image. Maybe the configuration is corrupted, or maybe it just stopped working. This exercise helps you determine why the server is not starting.

Before you start exploring some server start failures, it is useful to know just what happens when a server is started. When you type the command `startServer server1`, two JVMs are started. The first JVM is the Systems Management server launch utility. Its job is to locate the appropriate configuration (`server.xml`) and create the second JVM, which is the actual server process. Some default properties are set in the various `systemlaunch.properties` files. The creation of the second JVM is achieved by constructing a command (for example, `java -Dxxx -classpath x;y;z`) and executing it. This process that creates the server waits until it receives back status from the server process, unless you specified the `-nowait` parameter.

Requirements

Any additional software or tools are provided in the `Student` directory on your computer, or they are part of the computer setup process.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Prepare your lab image environment

In this section, you enable the text log for server1, and then ensure that all servers are stopped.

__ 1. Make sure that the deployment manager, node agent, and server1 are running. Start them if necessary.

__ 2. Enable HPEL logging.

- __ a. Log on to the administrative console as follows:
  - User name: `wasadmin`
  - Password: `web1sphere`
- __ b. Select **Troubleshooting > Logs and trace > server1**.
- __ c. Click **Switch to HPEL Node**.
- __ d. Click **OK** and **Save** the changes.
- __ e. Restart **server1**.

__ 3. Enable the text log for server1. In a previous exercise, you enabled HPEL for server1 but disabled the text log. For this exercise, it is easier to use the text log.

- __ a. Select **Troubleshooting > Logs and trace > server1**.
- __ b. Click **Configure HPEL text log**.
- __ c. Check the box for **Enable text log**.
- __ d. Click **OK** and **Save** the changes.
- __ e. Wait for the nodes to synchronize, and click **OK**.
- __ f. Log out of the administrative console.

__ 4. Stop server1.

- __ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`
- __ b. Enter the command: `./stopServer.sh server1 -user wasadmin -password web1sphere`

__ 5. Stop the node agent.

- __ a. Enter the command: `./stopNode.sh -user wasadmin -password web1sphere`

__ 6. Stop the deployment manager.

- __ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`
- __ b. Enter the command: `./stopManager.sh -user wasadmin -password web1sphere`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 2: Simulate and troubleshoot an application server start failure

In this scenario, you examine an application server start failure that results from the fact that the node agent of the server is not running.

__ 1. Delete any old TextLog files and the `startServer.log` for server1.

__ a. Using the file browser, navigate to the directory: `<profiles_root>/profile1/logs/server1`

__ b. Delete all the `TextLog_<time_stamp>.log` files if there are any.

__ c. Delete the `startServer.log` file.

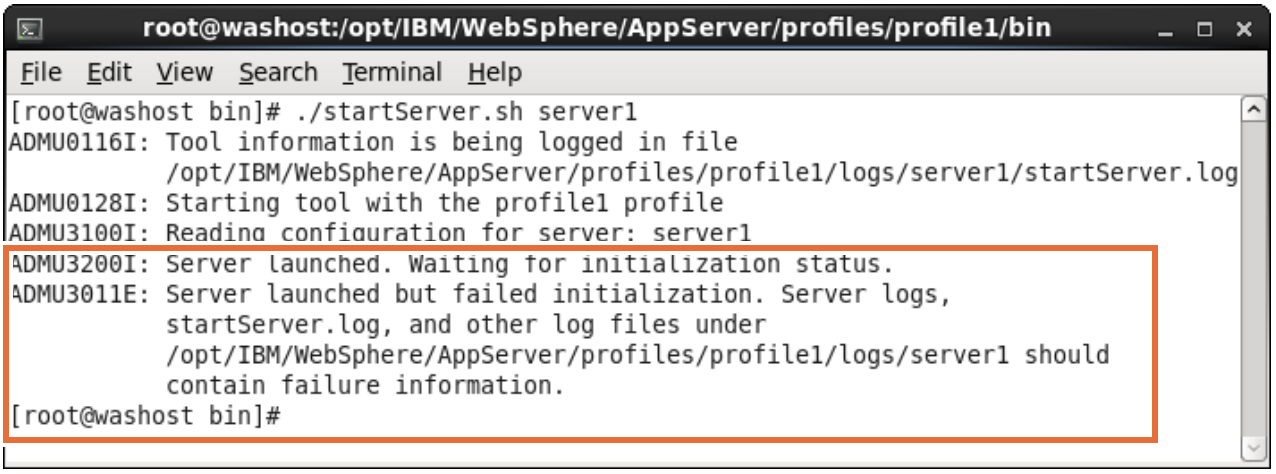
__ 2. Start server1 from the command line.

__ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./startServer.sh server1`

__ c. You see a message similar to:

```
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3011E: Server launched but failed initialization. Server logs,
           startServer.log, and other log files under
           opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1 should
           contain failure information.
[root@washost bin]#
```



__ 3. First, examine the exceptions in the `TextLog_<time_stamp>.log` file.

__ a. From the file browser, navigate to the logs directory of server1: `<profiles_root>/profile1/logs/server1`

__ b. Open `TextLog_<time_stamp>.log` with gedit.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Scroll down to the first error message and stack trace. You see an error message similar to this one:

```
WsServerImpl E WSVR0009E: Error occurred during startup
com.ibm.ws.exception.RuntimeError: com.ibm.ejs.EJSException: Could not register
with Location Service Daemon, which could only reside in the NodeAgent. Make sure
the NodeAgent for this node is up and running; nested exception is . . .
```

```
WsServerImpl  E   WSVR0009E: Error occurred during startup
com.ibm.ws.exception.RuntimeError: com.ibm.ejs.EJSException: Could not register with
Location Service Daemon, which could only reside in the NodeAgent. Make sure the
NodeAgent for this node is up and running.; nested exception is:
            org.omg.CORBA.ORBPackage.InvalidName:
LocationService:org.omg.CORBA.TRANSIENT: java.net.ConnectException: Connection
refused: connect:host=washost,port=9101  vmcid: IBM  minor code: E02  completed: No
            at com.ibm.ws.runtime.component.ORBImpl.start(ORBImpl.java:459)
            at
com.ibm.ws.runtime.component.ContainerHelper.startComponents(ContainerHelper.java:539)
            at
com.ibm.ws.runtime.component.ContainerImpl.startComponents(ContainerImpl.java:627)
...
```

__ d. Notice that the message shown accurately identifies the cause of the failure: the node agent is not running.

__ e. Close the `TextLog.log` file.

__ 4. Examine the messages in the `startServer.log`.

__ a. Open `startServer.log` with gedit.

__ b. Notice that there are a few messages in this log file. These messages are virtually what you saw in the command window when you entered the `start server` command with a few more trace events. However, the display of the current environment is worth examination. As you might recall from the lecture, there is information about the operating system, Java version, class path, and trace specification.

__ c. Now delete the `startServer.log` file and enter the `startServer` command again with the `-trace` option: `./startServer.sh server1 -trace`

__ d. Open the `startServer.log` file now, and you see the following message near the beginning of the file:

```
ManagerAdmin  I   TRAS0017I: The startup trace state is
*=info:com.ibm.*=all.
```

Information

All components are being traced at all detail levels so there is now more trace information in this log file. If for some reason the startup process fails before the `SystemOut.log` and `SystemErr.log` files were created, this trace information might be helpful for determining the cause of the start failure.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Scroll down through the trace events, and you can get an idea of the actions that occur during the server startup process.

__ f. When you are done examining the trace information, close the `startServer.log` file.

Information

The solution to this server start failure is fairly simple: Start the node agent first before attempting to start any application servers on the same node. However, it might be the case that the node agent is crashing or failing to start, and you must troubleshoot that issue.

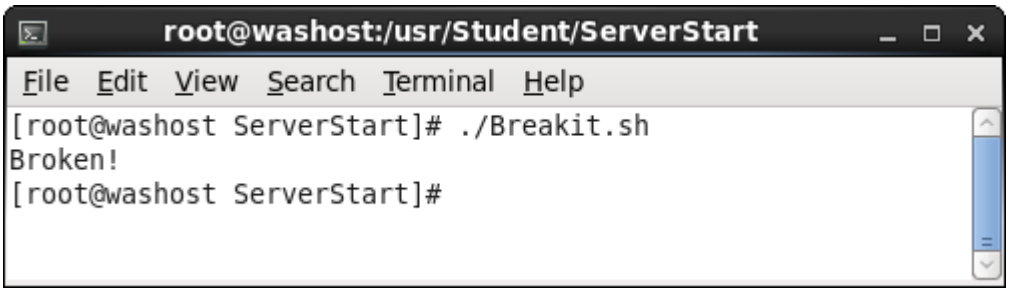
Section 3: Simulate and troubleshoot a node agent start failure

In this scenario, you simulate a node agent startup failure by making the `startup.jar` file in the `<WAS_HOME>/lib` directory unavailable. The node agent fails to start with a `NoClassDefFoundError` exception. You then troubleshoot the problem.

__ 1. Create the problem by renaming the `startup.jar` file.

__ a. From a Terminal window change directory to: `/usr/Student/ServerStart`

__ b. Enter: `./Breakit.sh`



This shell script runs the command to rename the `startup.jar` file to: `old_startup.jar`

__ 2. Delete the log files of the node agent.

__ a. Using the file browser, navigate to the directory: `<profiles_root>/profile1/logs/nodeagent`

__ b. Delete all files in this directory.

__ 3. Try to start the node agent.

__ a. From a Terminal window change directory to: `<profiles_root>/profile1/bin`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the command: `./startNode.sh -trace`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/profile1/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./startNode.sh -trace
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/profile1/logs/nodeagent/startServ
er.log
ADMU0128I: Starting tool with the profile1 profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3011E: Server launched but failed initialization. Server logs,
           startServer.log, and other log files under
           /opt/IBM/WebSphere/AppServer/profiles/profile1/logs/nodeagent should
           contain failure information.
[root@washost bin]#
```

__ c. Notice that the server process was launched but the initialization failed. Your next step is to examine the server log files if any were created.

__ 4. Examine the log file of the node agent.

__ a. Using the file browser, navigate to the directory: *<profiles_root>*/profile1/logs/nodeagent

__ b. You notice that both the `native_stderr.log` and the `startServer.log` contain messages. The `native_stdout.log` is empty.

__ c. Use gedit to open `native_stderr.log`.

__ d. You see a message similar to this one:

```
Exception in thread "main" java.lang.NoClassDefFoundError:
org.eclipse.core.launcher.Main
. . .
Caused by: java.lang.ClassNotFoundException: org.eclipse.core.launcher.Main
. . .
```

```
Exception in thread "main" java.lang.NoClassDefFoundError: org.eclipse.core.launcher.Main
        at com.ibm.wsspi.bootstrap.WSPreLauncher.launchEclipse(WSPreLauncher.java:371)
        at com.ibm.wsspi.bootstrap.WSPreLauncher.main(WSPreLauncher.java:142)
Caused by: java.lang.ClassNotFoundException: org.eclipse.core.launcher.Main
        at java.net.URLClassLoader.findClass(URLClassLoader.java:434)
        at java.lang.ClassLoader.loadClassHelper(ClassLoader.java:665)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:644)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:358)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:627)
        ... 2 more
```

__ e. The messages indicate that a class definition cannot be found, and a class is not found, possibly because the class is missing or its JAR file might be corrupted.

__ f. Close the `native_stderr.log` file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ g. Check the `startServer.log` for more details about the events that lead up to this exception. Open the `startServer.log` file with gedit.

__ h. Near the bottom of the startServer.log file, you see the command-line message:

```
ADMU3011E: Server launched but failed initialization...
```

__ i. Try reading through the trace events that lead up to this message. However, you probably might not find any information about the location of the missing class. It might be worthwhile to enable verbose class loading for the node agent and try starting it again.

__ j. Close the `startServer.log` file.

__ 5. Enable verbose class loading for the node agent. The safest way to do this task is to use the administrative console or a Jython script, but in this scenario the missing `startup.jar` file causes the deployment manager to fail on startup also. Therefore, the administrative console is not available, so you must manually edit the `server.xml` file of the node agent.

__ a. Use the file browser to navigate to the directory: `<profiles_root>/profile1/config/cells/washostCell/nodes/washostNode01/servers/nodeagent`

__ b. Open the `server.xml` file with gedit.

__ c. Search for the setting: `verboseModeClass="false"`

__ d. Change it to: `verboseModeClass="true"`

__ e. Save and close the `server.xml` file.

__ 6. Try again to start the node agent and generate verbose class loading output.

__ a. From a Terminal window, change to the directory: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./startNode.sh -trace`

__ c. From the file browser, navigate to: `<profiles_root>/profile1/logs/nodeagent`

__ d. Use gedit to open the `native_stderr.log` file.

__ e. Scroll down through the log, and you see all of the classes that are being loaded during the startup of the node agent. At the bottom of the log, you see the

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

NoClassDefFoundError exception again, but you also see what classes were loaded just before it.

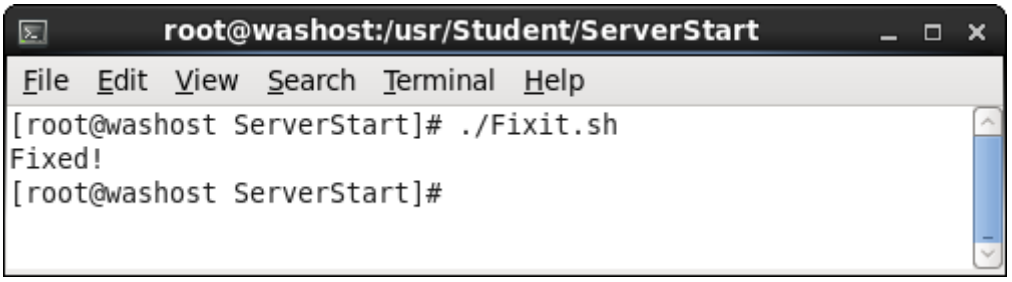
```
. . .
class load: org.eclipse.osgi.internal.resolver.StateWriter from:
file:/opt/IBM/WebSphere/AppServer/plugins/org.eclipse.osgi_.jar
class load: java/util/IdentityHashMap$1
class load: java/util/IdentityHashMap$IdentityHashMapIterator
class load: java/util/IdentityHashMap$1$1
class load: java/util/IdentityHashMap$IdentityHashMapEntry
class load: java/net/FileNameMap
class load: com/ibm/tools/attach/javaSE/AttachHandler$1
Exception in thread "main" java.lang.NoClassDefFoundError: org.eclipse.core.launcher.Main
                    at com.ibm.wsspi.bootstrap.WSPreLauncher.launchEclipse(WSPreLauncher.java:378)
                    at com.ibm.wsspi.bootstrap.WSPreLauncher.main(WSPreLauncher.java:150)
Caused by: java.lang.ClassNotFoundException: org.eclipse.core.launcher.Main
. . .
```

__ f. Though this information might not help pinpoint the exact cause of the **NoClassDefFoundError** exception, it might suggest that your Java libraries are corrupted. So your next step might be to restore the WebSphere Java libraries from system backups.

__ 7. Fix the problem by running a batch file that renames the `startup.jar` file correctly.

__ a. From a Terminal window, change directory to: `/usr/Student/ServerStart`

__ b. Enter: `./Fixit.sh`



This batch file runs the command to rename the `Old_startup.jar` file back to `startup.jar`.

__ 8. Verify that you can now start the node agent successfully.

__ a. From a Terminal window change directory to: *<profiles_root>*`/profile1/bin`

__ b. Enter the command: `./startNode.sh`

Wait for the node agent to start successfully.

__ 9. Disable verbose class loading for the node agent. Verbose class loading can slow the performance of the server, so it must be disabled after the diagnostic data is collected.

__ a. Use the file browser to navigate to the directory: *<profiles_root>*`/profile1/config/cells/washostCell/nodes/washostNode01/servers/nodeagent`

__ b. Open the `server.xml` file with gedit.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Search for the setting: `verboseModeClass="true"`

__ d. Change it to: `verboseModeClass="false"`

__ e. Save and close the `server.xml` file.

Information

In this scenario, you explored another way that a server can fail to start: missing or corrupted Java classes that the Classloader of the server needs to load at startup time. Although you simulated this problem by explicitly renaming a JAR file, making its classes invisible to the Classloader, this type of problem is not uncommon. The file system can get corrupted, or restoring from backups might corrupt or misplace directories.

Section 4: Simulate and troubleshoot a deployment manager start failure

In this scenario, you simulate a deployment manager start failure that a port conflict causes. You configure another application to use the SOAP port of the deployment manager. After the deployment manager fails to start, you will troubleshoot the problem.

__ 1. Delete the logs of the deployment manager.

__ a. Using the file browser, navigate to the directory: `<profiles_root>/Dmgr/logs/dmgr`

__ b. Delete all files in this directory.

__ 2. Create the port conflict. You configure the IBM HTTP Administration on your system to run on port 8879.

__ a. If IBM HTTP Administration Server is running, stop it. From a Terminal window, change directory to: `/opt/IBM/HTTPServer/bin`

__ b. Enter the command: `./adminctl stop`

__ c. Change to the directory: `/opt/IBM/HTTPServer/conf`

__ d. Use gedit to open the `admin.conf` file.

__ e. Search for `Listen 8008` and change the port to `8879`

__ f. Save and close the `admin.conf` file.

__ g. Start IBM HTTP Administration Server. From a Terminal window, change directory to: `/opt/IBM/HTTPServer/bin`

__ h. Enter the command: `./adminctl start`

__ 3. Start the deployment manager.

__ a. From a Terminal window, change directory to: `<profiles_root>/Dmgr/bin`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the command: `./startManager.sh -trace`

```
root@washost:/opt/IBM/WebSphere/AppServer/profiles/Dmgr/bin
File  Edit  View  Search  Terminal  Help
[root@washost bin]# ./startManager.sh -trace
ADMU0116I: Tool information is being logged in file
           /opt/IBM/WebSphere/AppServer/profiles/Dmgr/logs/dmgr/startServer.log
ADMU0128I: Starting tool with the Dmgr profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3028I: Conflict detected on port 8879.  Likely causes: a) An instance of
           the server dmgr is already running  b) some other process is using
           port 8879
ADMU3029I: Conflict detected on port 8879 for endpoint SOAP_CONNECTOR_ADDRESS
           of the server dmgr
ADMU3027E: An instance of the server may already be running: dmgr
ADMU0111E: Program exiting with error:
           com.ibm.websphere.management.exception.AdminException: ADMU3027E: An
           instance of the server may already be running: dmgr
ADMU0211I: Error details may be seen in the file:
           /opt/IBM/WebSphere/AppServer/profiles/Dmgr/logs/dmgr/startServer.log
[root@washost bin]#
```

__ c. Notice that the conflict with port 8879 is detected, and the likely causes are listed as:

- An instance of the server dmgr is already running
- Some other process is using port 8879

Next, you investigate each of these cases.

__ 4. Check whether another instance of the deployment manager is already running.

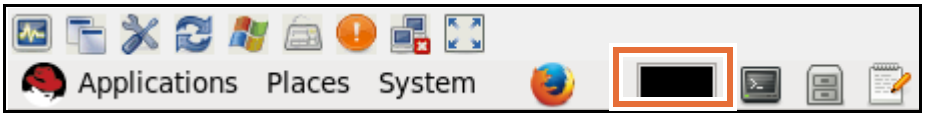
Information

One way to check for a running instance is to look in the logs file of the deployment manager, `<profiles_root>/Dmgr/logs/dmgr`, for the `dmgr.pid` file. If this file exists, open it to find the PID.

Use the System Monitor to see whether a `Java` process is running with this PID. However, this file does not exist for this lab since you previously deleted the contents of the logs directory.

__ a. Use the System Monitor to look for `Java` processes. If any are listed, note their PID and try to identify them using the PID file for the other WebSphere servers that might be running, such as: server1, node agent, deployment manager.

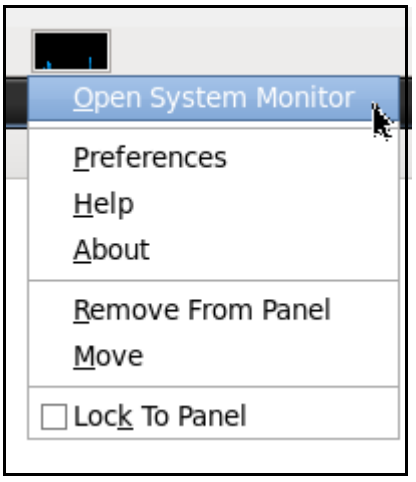
__ b. Start the System Monitor by right-clicking the desktop icon.



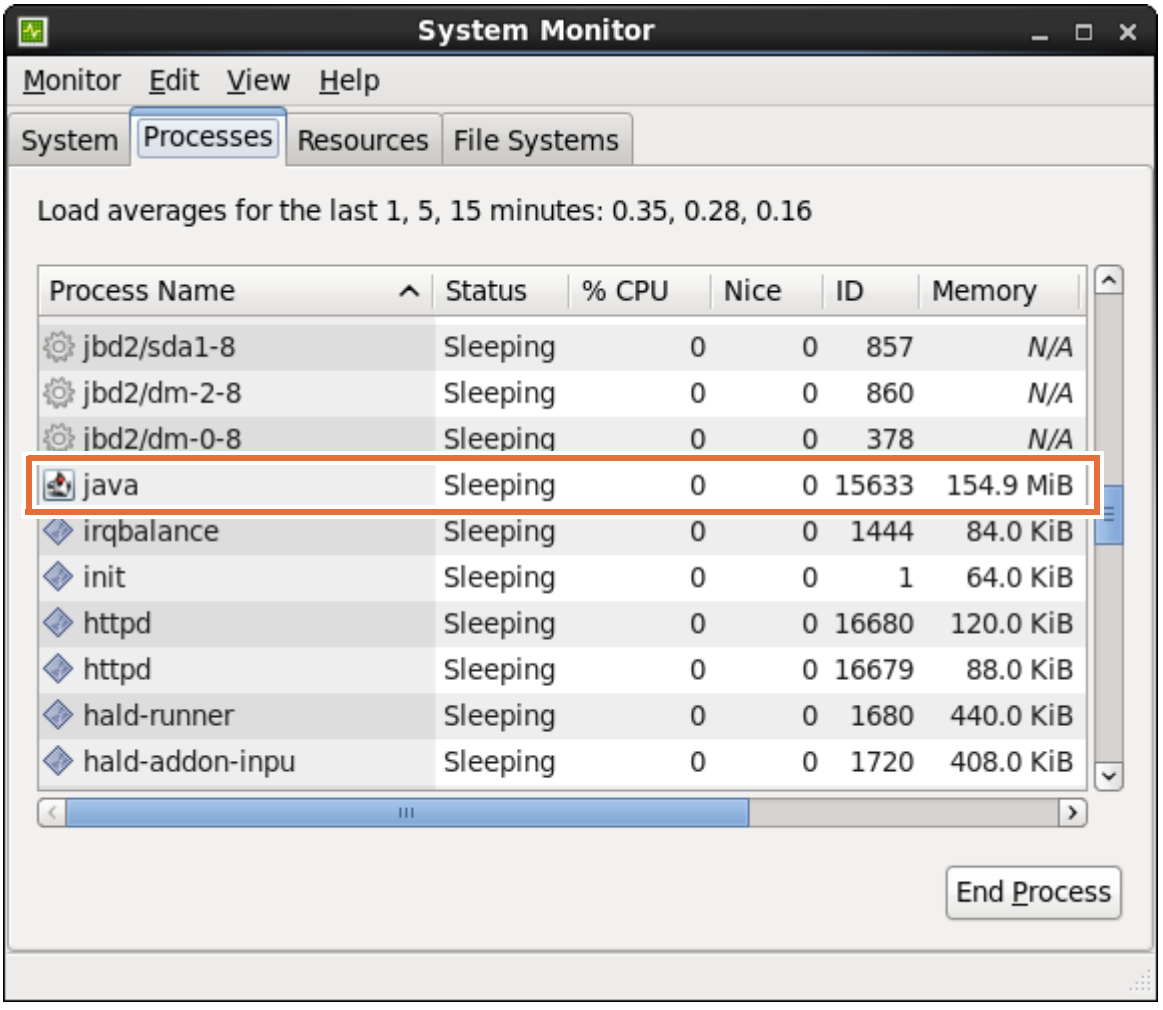
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select **Open System Monitor**.



__ d. When the System Monitor opens, select the **Processes** tab and scroll down to find any `java` processes.



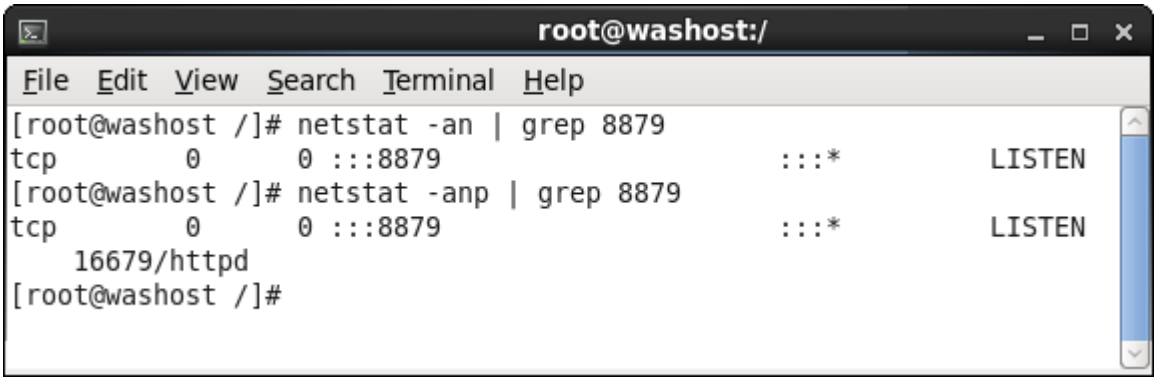
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. You might not see any Java processes, but for illustrative purposes, the node agent is still running from the previous section. However, if you do see a Java process, and the PID matches the value the `dmgr.pid` file, you would confirm the fact that an instance of the deployment manager is already running.

__ f. Close the System Monitor.

__ 5. Use the system `netstat` command to check whether some other process is using port 8879.

__ a. From a Terminal window, enter the command: `netstat -an | grep 8879`



__ b. Notice that the `netstat` command reveals that some process is listening on port 8879. If you use the `-anp` options, users can see what process is listening on that port. In this scenario, you see that an `httpd` process is using port 8879.

Information

On UNIX systems, the `netstat` and `lsof -i` commands can be used to identify the process ID associated with each open port. For the Windows operating system, various port scanner tools can be downloaded. You can use the CurrPorts tool from `www.nirsoft.net`.

Also, the IBM Port Scanning tool in the IBM Support Assistant V4.1 can be used to identify which processes are listening on which ports.

__ 6. Examine the `startServer.log` file.

__ a. Using the file browser, navigate to the directory *<profiles_root>*`/Dmgr/logs/dmgr`.

__ b. You might notice that *only* the `startServer.log` file was created, indicating that the Launch utility never created the server process; therefore the server initialization process never began. Because you were instructed to use the `-trace` option to start the server, there is extensive trace output in the `startServer.log` file.

__ c. Open the `startServer.log` file with gedit and read through the trace information. Again, this trace output allows you to follow the actions that occurred during the server startup.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. When you are done examining the `startServer.log` file, close it.

Information

Now that the source of the port conflict is identified, there are a number of ways to resolve the conflict. The port for either the IBM HTTP Administration Server or the deployment manager might be changed, or you can stop the IBM HTTP Administration Server if necessary. For this exercise, you change the port of the IBM HTTP Administration Server.

__ 7. Change the port of the IBM HTTP Administration.

__ a. Stop the IBM HTTP Administration Server. From a Terminal window, change directory to: `/opt/IBM/HTTPServer/bin`

__ b. Enter the command: `./adminctl stop`

__ c. From a Terminal window, navigate to the directory `/opt/IBM/ HTTPServer/conf`.

__ d. Use gedit to open the `admin.conf` file.

__ e. Search for `Listen 8879` and change the port back to `8008`.

__ f. Save and close the `admin.conf` file.

__ g. Start IBM HTTP Administration Server. From a Terminal window, change directory to: `/opt/IBM/HTTPServer/bin`

__ h. Enter the command: `./adminctl start`

__ 8. Verify that IBM HTTP Administration is not using port 8879.

__ a. From a Terminal window, enter the command: `netstat -an | grep 8879`

__ b. Verify that no process is using port 8879.

__ 9. Verify that you can now start the deployment manager without a port conflict.

__ a. From a Terminal window, enter the command: `./startManager.sh -trace`

__ b. Confirm that the deployment manager now starts *without* a port conflict.

End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

The application server failure resulted from the fact that the node agent of server1 was not running. You explored this case by examining log files and by tracing the startup of the server.

The node agent failure was caused by renaming the `startup.jar` file in `<WAS_HOME>/lib`. You explored this case by examining log files and traces. You enabled verbose class loading and examined the verbose output for clues as to which classes were not successfully loaded. Finally, the problem was resolved by renaming the `startup.jar` file.

The deployment manager failure was the result of a port conflict. To create the port conflict, IBM HTTP Administration was reconfigured to use port 8879. This case was explored by using the operating system tool, `netstat`, to determine which ports were being used, and by using a port scanning tool to determine which process was using port 8879. Finally, you reconfigured IBM HTTP Administration to use the default port, and the problem was resolved.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 11.Troubleshooting request flow and web container problems

What this exercise is about

This exercise demonstrates what web resource request problems look like to the web user when the application server is not responding or the requested resource is not available.

What you should be able to do

At the end of the exercise, you should be able to:

- Configure web server logs for tracing
- Configure HTTP plug-in tracing
- Examine trace logs manually
- Use the WebContainerDP diagnostic provider
- Examine plug-in trace logs by using the Web Server Plug-in Analyzer tool

Introduction

In this exercise, you examine requests for various web resources, including HTML files and JSPs. You verify how each component in the request flow handles the request by examining log files. You enhance the information that is written to these log files when you configure tracing on different components such as the WebSphere HTTP server plug-in and the application server.

Requirements

This lab exercise requires that the PlantsByWebSphere application is installed on the application server server1. This lab also requires that the lab computers run an external HTTP server with a WebSphere plug-in.

- r
-

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Prepare the environment and change log levels

You need to complete several steps to prepare your environment for the exercises that follow. Among other things, you need to configure log level to trace for the HTTP server and the plug-in. You also need to make sure the plug-in is configured with the PlantsByWebSphere web modules.

__ 1. Ensure that the deployment manager is started.

- __ a. From a Terminal window, change the directory to: `<profiles_root>/Dmgr/bin`
- __ b. Enter the command `./serverStatus.sh dmgr -user wasadmin -password web1sphere`
- __ c. If dmgr is Stopped, start it with: `./startManager.sh`

__ 2. Ensure that the node agent is running.

- __ a. From a Terminal window, change directory to: `<profiles_root>/profile1/bin`
- __ b. Enter the command `./serverStatus.sh -all -user wasadmin -password web1sphere`
- __ c. If the node agent is stopped, start it with `./startNode.sh` and rerun the `./severStatus.sh` command after the `./startNode.sh` command completes.

__ 3. Start the HTTP Administration server if it is not already running. Since the IBM HTTP Server is configured as webserver1 on an unmanaged node in WebSphere, the HTTP Administration server must be running to manage webserver1 from the administrative console.

- __ a. From a Terminal window, change directory to: `/opt/IBM/HTTPServer/bin`
- __ b. Enter the command: `./adminctl start`

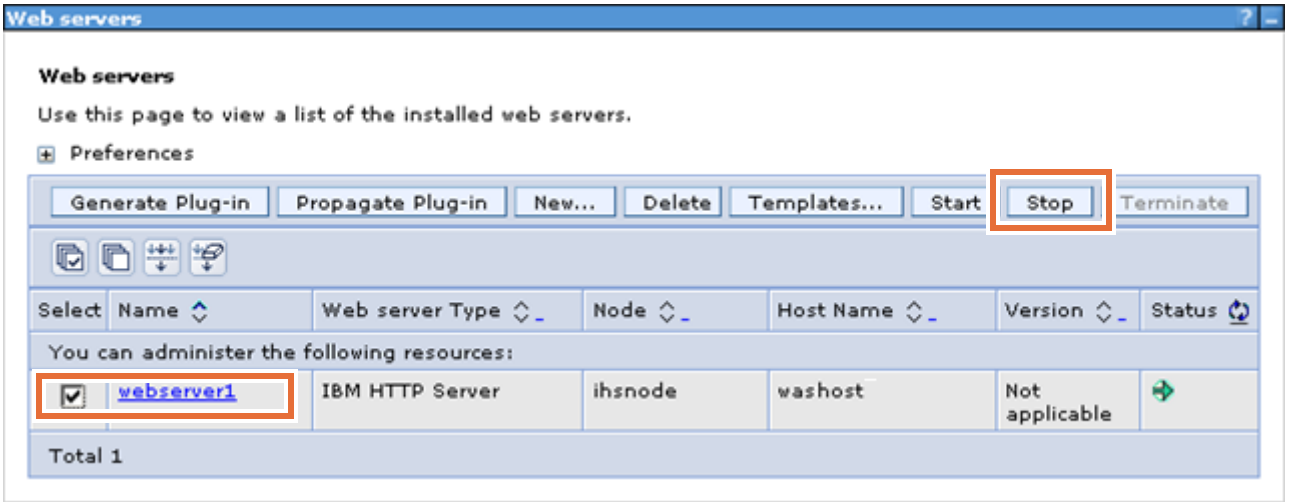
__ 4. Stop the managed HTTP server webserver1.

- __ a. Log on to the administrative console as follows:
  - User ID: `wasadmin`
  - Password: `web1sphere`
- __ b. Click **Servers > Server Types > Web servers**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. If it is running, select **webserver1** and click **Stop**.



__ d. Wait for the server to stop.

__ 5. Clear the HTTP server log files.

__ a. From a file browser, navigate to `/opt/IBM/HTTPServer/logs`. You can use a terminal window instead of the file browser.

__ b. Delete the following files. Do not delete the folders named `postinstall` and `config`.

- `admin_access_log`
- `admin.pid`
- `access_log`
- `admin_error.log`
- `error_log`

Note

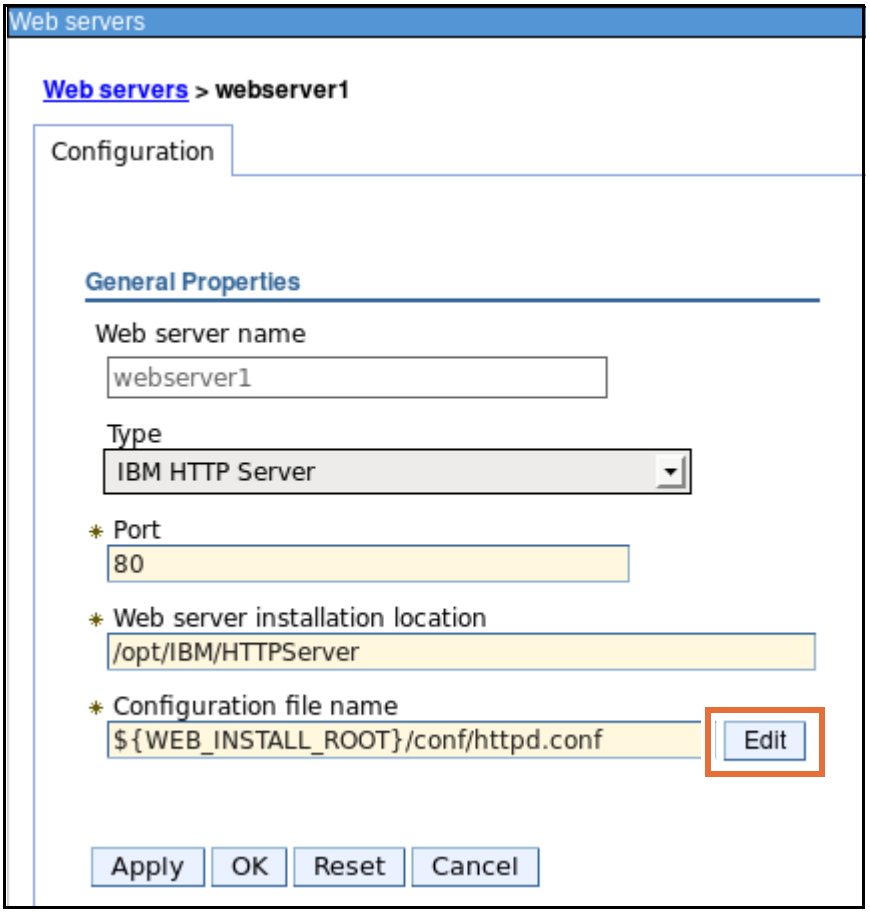
In a real testing or production environment, it is considered a good practice to archive the log files before deleting them.

__ 6. Change the log level for the HTTP server.

__ a. From the administrative console, select **Servers > Server Types > Web servers > webserver1**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Click **Edit** beside the Configuration file name field.



__ c. Search for the `LogLevel` directive and change it to: `LogLevel` debug

```
#
# LogLevel: Control the number of messages logged to the error log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
LogLevel debug
```

__ d. Click **OK**.

__ 7. Start the managed HTTP server webserver1.

__ a. Click **Servers > Server Types > Web servers**.

__ b. Select **webserver1** and click **Start**.

__ c. Wait for the server to start.

__ 8. Change the log level for the WebSphere HTTP server plug-in to Trace.

__ a. Click **Servers > Server Types > Web servers**.

__ b. Click the link for **webserver1**.

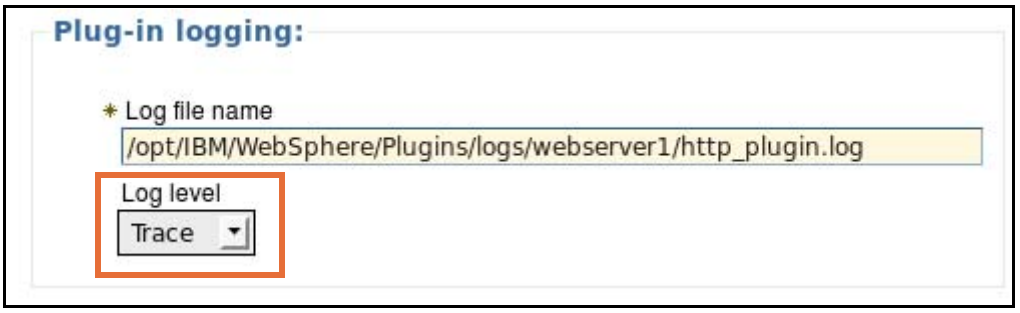
__ c. Under **Additional Properties**, click **Plug-in properties**.

__ d. Scroll down and locate **Plug-in logging:**

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Change the **Log level** to **Trace**.



__ f. Click **OK**.

__ g. **Save** your changes to the master configuration. Wait for the synchronization process to complete. Click **OK**.

__ 9. Restart the HTTP server.

__ a. From the administrative console, select **Servers > Server Type > Web servers**.

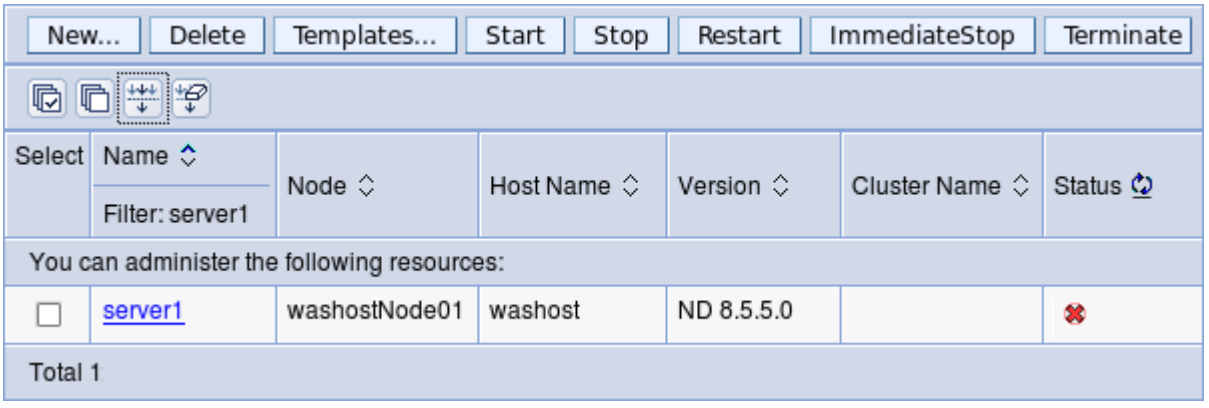
__ b. Select **webserver1** and click **Stop**.

__ c. Select **webserver1** and click **Start**.

__ d. Wait for the server to start.

__ 10. Verify that server1 is not running.

__ a. From the administrative console, select **Servers > Server Type > WebSphere application servers**.



__ b. Confirm that the status of server1 is **Stopped**. If it is not stopped, select **server1** and click **Stop**.

__ 11. Map the PlantsByWebSphere application to the HTTP server.

__ a. From the administrative console, select **Applications > Application Types > WebSphere enterprise applications**.

__ b. Click the link for **PlantsByWebSphere**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. In **Modules** section, click the **Manage Modules** link. The **Manage Modules** panel is shown. Examine the list of mappings. Ensure that the **Module** entries are mapped to the targets, which are identified under **Server**. The correct mapping is shown.

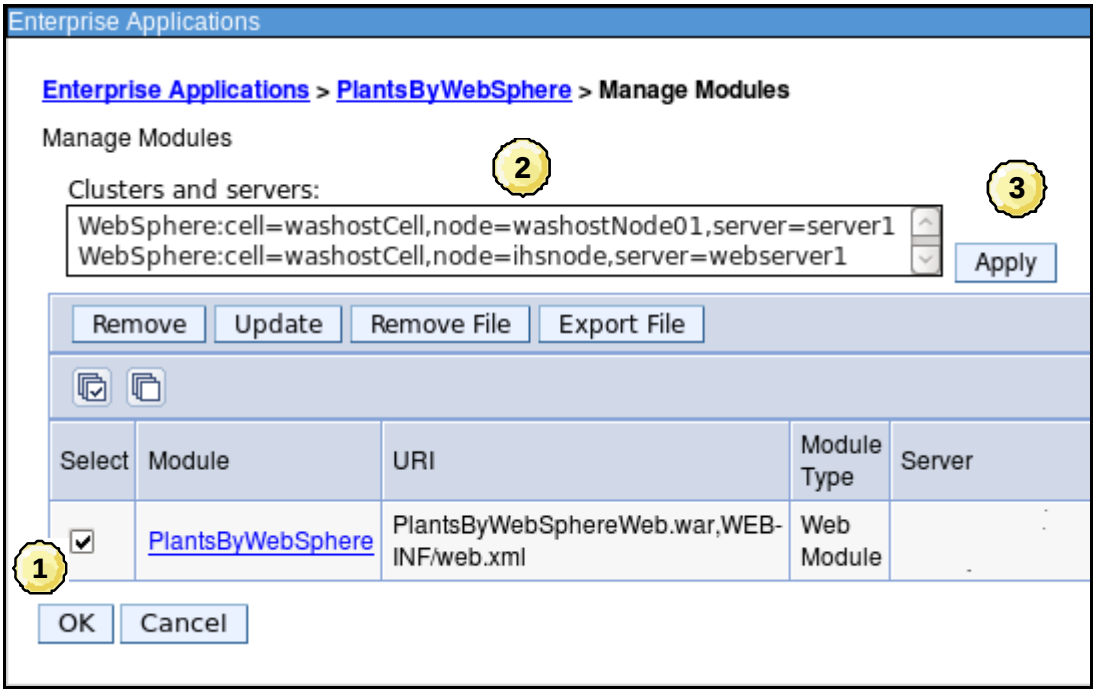
Module Type	Server
Web Module	WebSphere:cell=washostCell,node=ihsnode,server=webserver1 WebSphere:cell=washostCell,node=washostNode01,server=server1

__ d. The module must be mapped to both server1 and webserver1. If the module is not mapped to both server1 and webserver1 as shown, change the mappings as needed.

1) Select the module that you want mapped to the same targets. In the list of mappings, place a check mark in the **Select** check boxes beside the module.

2) From the **Clusters and Servers** list, select one or more targets. Use the **Ctrl** key to select multiple targets. For example, to have a web server serve your application, use the **Ctrl** key to select an application server or cluster and the web server together. This action generates the plug-in configuration file, `plugin-cfg.xml`, for that web server, which is based on the applications that are routed through it.

3) Click **Apply**.



__ e. Verify that the module is mapped to both **webserver1** and **server1**.

__ f. Click **OK**.

__ g. **Save** the changes to the master configuration.

 © Copyright IBM Corp. 2013

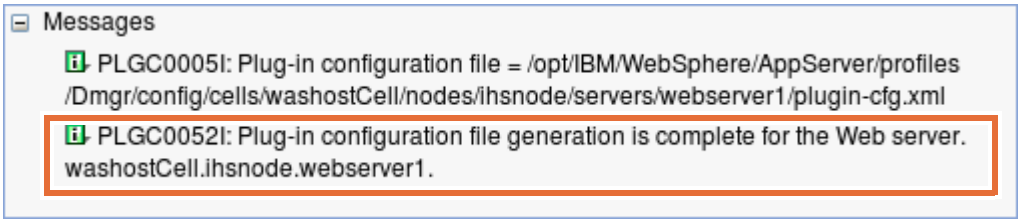
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 12. Regenerate and propagate the HTTP plug-in. By default, a web server is configured to automatically generate the plug-in every 60 seconds; however, you manually complete this step in this exercise.

__ a. From the administrative console, select **Servers > Server Types > Web servers**.

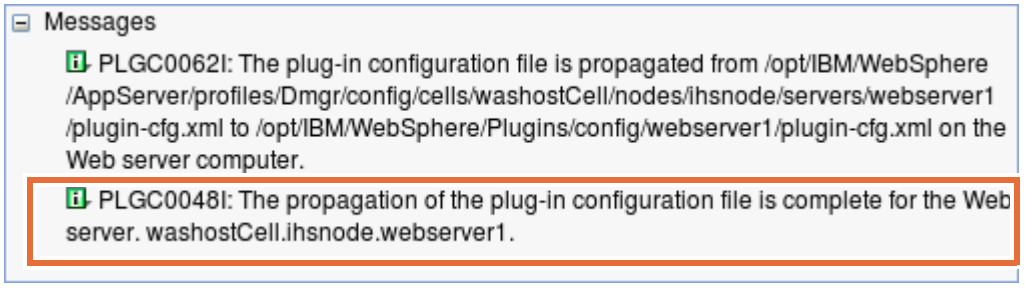
__ b. Select **webserver1** and click **Generate Plug-in**.

__ c. Wait for the plug-in generation to complete.



__ d. Select **webserver1** and click **Propagate Plug-in**.

__ e. Wait for the plug-in propagation to complete.



Section 2: Examine the results of a successful web request

In this section, you verify that your local HTTP server can successfully serve a request for static content from its own local document directory.

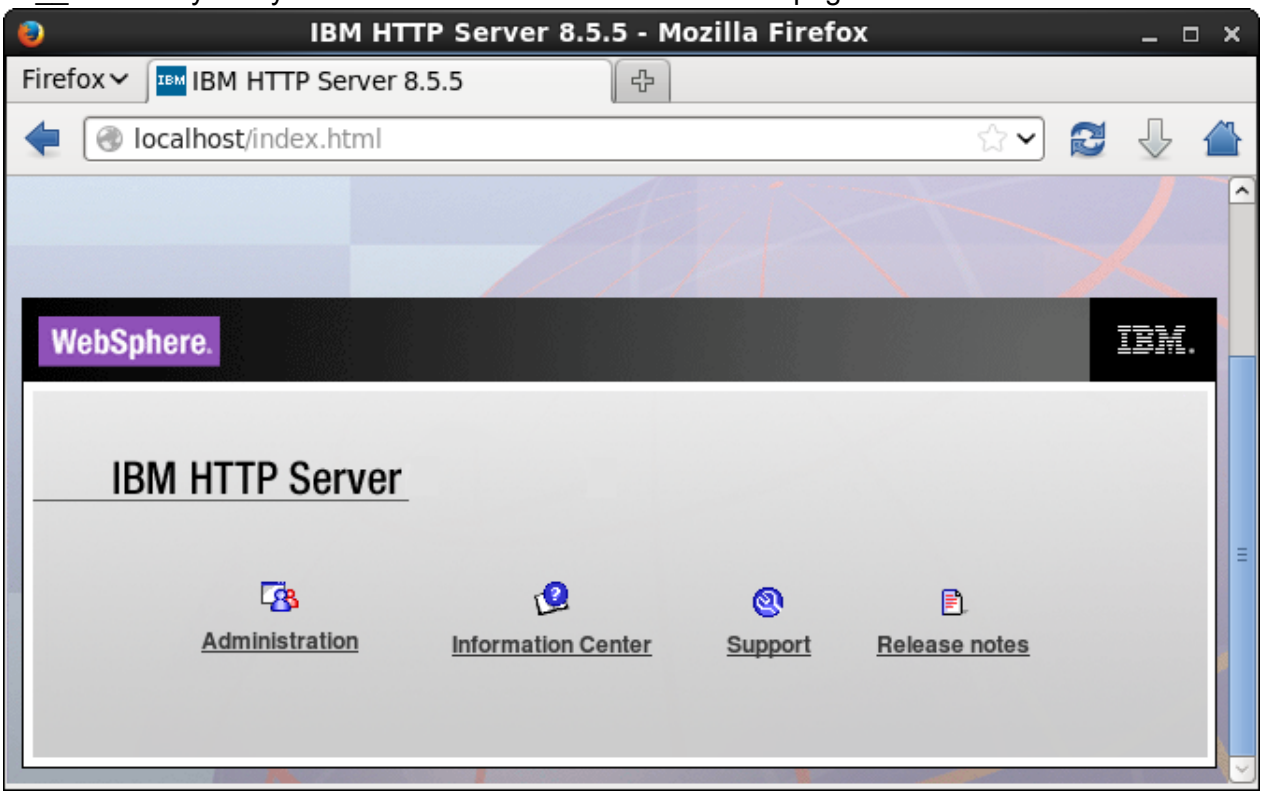
__ 1. Submit a request from the browser for a web page that the HTTP server serves.

__ a. Start Firefox.

__ b. Type `http://localhost/index.html` in the address bar.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Verify that you see the IBM HTTP Server welcome page.



__ d. Close the browser.

__ 2. Examine the HTTP log files of the server.

__ a. From the file browser or from the command line, navigate to: `/opt/IBM/HTTPServer/logs`

__ b. Use gedit to open the `access.log` file, or from the command line you can use vi or emacs. You see messages similar to:

```
127.0.0.1 - - [05/Feb/2014:09:22:29 -0500] "GET /index.html HTTP/1.1" 200 3493
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/odot.gif HTTP/1.1" 200 49
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/administration.gif
HTTP/1.1" 200 223
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/support.gif HTTP/1.1" 200
150
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /http_server_styles.css HTTP/1.1"
200 1184
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/background.gif HTTP/1.1"
200 183099
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/foreground.gif HTTP/1.1"
200 52919
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/notes.gif HTTP/1.1" 200 170
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/favicon.ico HTTP/1.1" 200
318
127.0.0.1 - - [05/Feb/2014:09:22:30 -0500] "GET /images/help.gif HTTP/1.1" 200 210
```

__ c. Notice that the HTTP response code is 200 for the `index.html` and the `.gif` files.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Note

If you see a 304 code instead of 200, then your browser is probably serving this resource from its cache. Clear the browser cache and try again.

__ d. Close the `access.log` file and exit the browser.

Section 3: Examine the results of a failed web request

In this section, you look at the information the HTTP server writes to its local log files when a request for a resource fails.

__ 1. Submit a request from the browser for a web page that the HTTP server does **not** serve.

__ a. Start a new browser window.

__ b. Type the following URL in the address bar: `http://localhost/index99.html`

__ c. Notice that this request results in an error page and a 404 response code.

__ 2. Examine the HTTP log files of the server.

__ a. Navigate to: `/opt/IBM/HTTPServer/logs/`

__ b. Open the `access.log` file. A message similar to this one is recorded:

```
127.0.0.1 - - [05/Feb/2014:09:27:26 -0500] "GET /index99.html HTTP/1.1" 404 271
127.0.0.1 - - [05/Feb/2014:09:27:26 -0500] "GET /favicon.ico HTTP/1.1" 404 270
127.0.0.1 - - [05/Feb/2014:09:27:26 -0500] "GET /favicon.ico HTTP/1.1" 404 270
```

__ c. Notice that the HTTP response code is 404, the File Not Found code.

__ d. Close the `access.log` file.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Open the `error.log` file. At the bottom of the file, you see a message similar to this one:

```
[Wed Feb 05 09:27:26 2014] [error] [client 127.0.0.1] File does not exist:
/opt/IBM/HTTPServer/htdocs/index99.html
[Wed Feb 05 09:27:26 2014] [error] [client 127.0.0.1] File does not exist:
/opt/IBM/HTTPServer/htdocs/favicon.ico
[Wed Feb 05 09:27:26 2014] [error] [client 127.0.0.1] File does not exist:
/opt/IBM/HTTPServer/htdocs/favicon.ico
```

Information

The cause of this failure is attempting to request a page that does not exist within the document root directory of the web server.

The access log has a message similar to:
`127.0.0.1 - - [<Time Stamp>] 'GET /index99.html HTTP/1.1' 404 280`.

The error log has a related message similar to:

```
[<Time Stamp>] [error] [client 127.0.0.1] File does not exist:
/opt/IBM/HTTPServer/htdocs/index99.html
```

Two diagnostic items can be gathered from the error log message:

- The requested file name
- The exact directory from which the web server attempted to serve the content

__ f. Close the `error.log` file and exit the browser.

Section 4: Examine failures at the plug-in

What happens when the HTTP server plug-in determines that a request must be forwarded to an application server but the plug-in is unable to contact an application server? In this part of the exercise, you look at the information that the HTTP server plug-in writes to its log file for this set of circumstances.

__ 1. Clear the `http_plugin.log` file.

__ a. Use the administrative console to stop webserver1.

__ b. Navigate to the directory:

`/opt/IBM/WebSphere/Plugins/logs/webserver1`

__ c. Rename the existing `http_plugin.log` file to: `http_plugin_old.log`

__ d. Use the administrative console to start **webserver1**.

__ e. Navigate to the directory: `/opt/IBM/WebSphere/Plugins/logs/webserver1`
Notice that a new `http_plugin.log` file is created in the logs directory.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Submit a request from the browser for a page that the application server serves.

__ a. Start a new Firefox window.

__ b. Type `http://localhost/PlantsByWebSphere` in the address bar.

__ c. Notice that you get an **Internal Server Error** page.

__ d. Close the browser.

__ 3. Examine the HTTP `access.log` of the server.

__ a. Navigate to the `/opt/IBM/HTTPServer/logs` directory.

__ b. Open the `access.log` file. You see a message similar to this message:

```
127.0.0.1 - - [05/Feb/2014:09:42:05 -0500] "GET /PlantsByWebSphere HTTP/1.1" 500
597
```

__ c. Notice that the HTTP response code is 500, the Internal Server Error code.

__ d. Close the `access.log` file.

__ 4. Examine the `http_plugin.log` file.

__ a. Use gedit to open the file:
`/opt/IBM/WebSphere/Plugins/logs/webserver1/http_plugin.log`

__ b. Select **Search > Find** to search for the following string:

`WebSphere will handle: /PlantsByWebSphere`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Find the following message:

```
[05/Feb/2014:09:42:05.39463] 000030a3 323de710 - DEBUG: mod_was_ap22_http:
as_translate_name: WebSphere will handle: /PlantsByWebSphere (client port is 60554)
[05/Feb/2014:09:42:05.39475] 000030a3 323de710 - DEBUG: mod_was_ap22_http:
as_handler: In the app server handler
[05/Feb/2014:09:42:05.39479] 000030a3 323de710 - DEBUG: ws_common:
ws_handle_request: Handling WebSphere request
[05/Feb/2014:09:42:05.39480] 000030a3 323de710 - TRACE: ws_common:
websphereBeginRequest: Beginning a client request
[05/Feb/2014:09:42:05.39482] 000030a3 323de710 - DEBUG: ws_common:
websphereCheckConfig: Current time is 1391611325, next stat time is 1391611385
```

Note

The message indicates that the plug-in was able to successfully determine that WebSphere handles the request.

__ d. Select **Search > Find** to search for the following string:

`websphereFindTransport`

__ e. Find these messages:

```
[05/Feb/2014:09:42:05.39676] 000030a3 323de710 - TRACE: ws_common:
websphereFindTransport: Finding the transport for server 0x729310
[05/Feb/2014:09:42:05.39677] 000030a3 323de710 - TRACE: ws_common:
websphereFindTransport: Inbound is SSL:0 httptransports=1, https_transports=1
[05/Feb/2014:09:42:05.39680] 000030a3 323de710 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): washost on port 9080
[05/Feb/2014:09:42:05.39683] 000030a3 323de710 - DEBUG: ws_common:
websphereExecute: Executing the transaction with the app server reqInfo is 10000df0
useExistingStream=0, client->stream=(nil)
[05/Feb/2014:09:42:05.39684] 000030a3 323de710 - DEBUG: ws_common:
websphereGetStream: Getting the stream to the app server (keepalive 28)
```

Note

These messages show that the plug-in determined which transport should forward the request.

__ f. Select **Search > Find** to search for the strings:

`OS err = 107`

and

`Failed to find an app server`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

and

`Failed to handle request`

__ g. Find the following messages:

```
[05/Feb/2014:09:42:05.40057] 000030a3 323de710 - ERROR: Connection to washost:9080
failed
[05/Feb/2014:09:42:05.40062] 000030a3 323de710 - ERROR: ws_common:
websphereGetStream: socket error conditions pending, OS err = 107
[05/Feb/2014:09:42:05.40065] 000030a3 323de710 - DEBUG: ws_common:
websphereGetStream: socket 17 closed - failed non-blocking connect
[05/Feb/2014:09:42:05.40067] 000030a3 323de710 - ERROR: ws_common:
websphereExecute: Failed to create the stream
[05/Feb/2014:09:42:05.40069] 000030a3 323de710 - DEBUG: ws_server:
serverSetFailoverStatus: Request to mark
 washostNode01_server1 down ignored.
[05/Feb/2014:09:42:05.40070] 000030a3 323de710 - STATS: ws_server:
serverSetFailoverStatus: Server washostN
ode01_server1 : pendingRequests 0 failedRequests 1 affinityRequests 0 totalRequests
0.[05/Feb/2014:09:42:05.40071] 000030a3 323de710 - ERROR: ws_common:
websphereHandleRequest: Failed to execute the transaction to
'washostNode01_server1'on host 'washost'; will try another one
[05/Feb/2014:09:42:05.40073] 000030a3 323de710 - ERROR: ws_common:
websphereWriteRequestReadResponse: Failed to find an app server to handle this
request
[05/Feb/2014:09:42:05.40074] 000030a3 323de710 - ERROR: ESI: getResponse: failed to
get response: rc = 2
[05/Feb/2014:09:42:05.40076] 000030a3 323de710 - DEBUG: ESI: esiHandleRequest:
failed to get response
[05/Feb/2014:09:42:05.40077] 000030a3 323de710 - DEBUG: ESI:
esiRequestUrlStackDestroy
[05/Feb/2014:09:42:05.40079] 000030a3 323de710 - DEBUG: ESI: cacheURL:
'/PlantsByWebSphere'
[05/Feb/2014:09:42:05.40080] 000030a3 323de710 - DETAIL: ESI: esiRequestPopUrl:
'/PlantsByWebSphere'
[05/Feb/2014:09:42:05.40081] 000030a3 323de710 - DEBUG: ESI: esiUrlDestroy:
'/PlantsByWebSphere'
[05/Feb/2014:09:42:05.40084] 000030a3 323de710 - ERROR: ws_common:
websphereHandleRequest: Failed to handle request
```

Information

These messages are not directly one line after the other. The `OS err=107` (Linux code) error message indicates that the plug-in was unable to successfully create a connection to the application server. In your lab scenario, there was no other server for the request to fail over to because server1 is not running.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

h. Close the `http_plugin.log` file.

Section 5: Examine a failed WebSphere Application Server request

In the following scenario, you explore what happens when the application server fails to respond to a request for a JSP. You configure tracing on the web container, create the request failure, and analyze the trace messages in the server log file.

__ 1. Start server1.

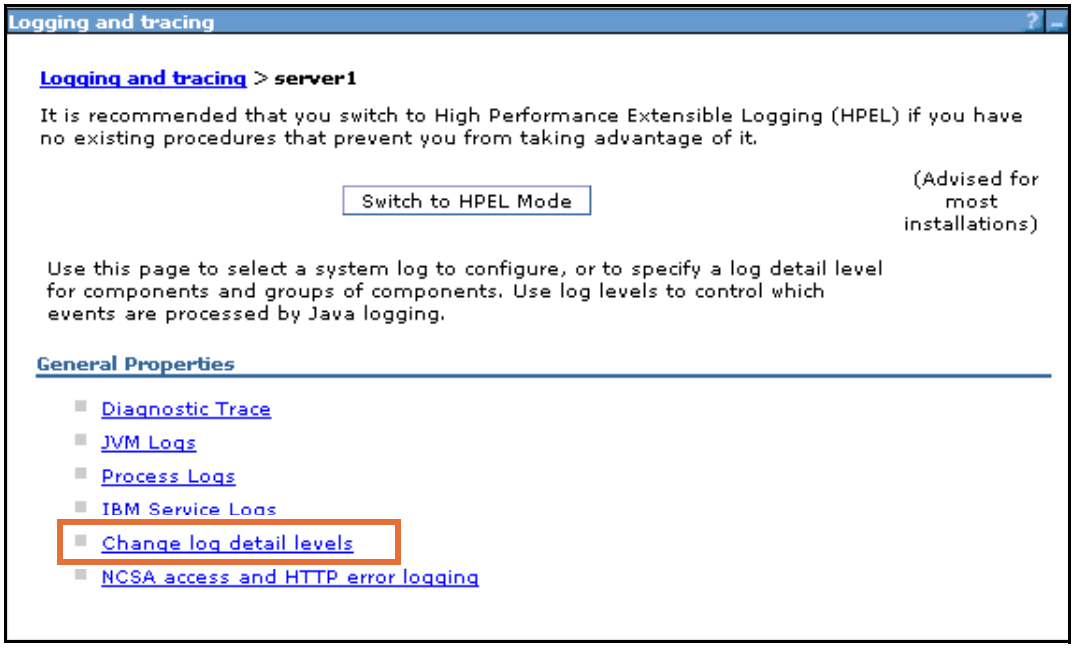
__ a. From the administrative console click **Servers > Server Types > WebSphere application servers**.

__ b. Check the box for **server1** and click **Start**.

__ c. Wait for server1 to start.

__ 2. Configure tracing on the web container of server1.

__ a. Select **Troubleshooting > Logs and trace > server1**.



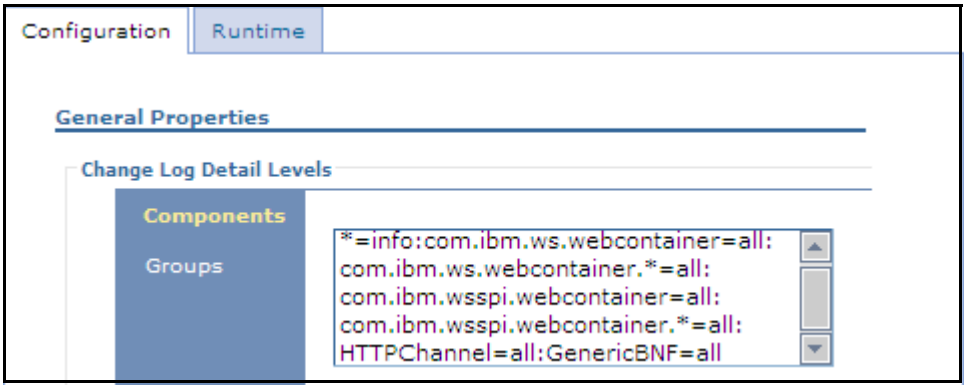
__ b. Click the link **Change Log Detail Levels**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

On the Configuration tab, under **Change Log Detail Levels**, enter the following trace string (you can copy and paste the trace string from the file: `/usr/Student/RequestFlow/TraceString.txt`)

```
*=info:com.ibm.ws.webcontainer=all:
com.ibm.ws.webcontainer.*=all:
com.ibm.wsspi.webcontainer=all:
com.ibm.wsspi.webcontainer.*=all:
HTTPChannel=all:GenericBNF=all
```



__ c. Click **OK**, and **Save** the changes to the master configuration.

__ d. Wait for the nodes to synchronize and click **OK**.

__ 3. Clear the `http_plugin.log` file.

__ a. Use the administrative console to stop **webserver1**.

__ b. Navigate to: `/opt/IBM/WebSphere/Plugins/logs/webserver1`

__ c. Rename the existing `http_plugin.log` file to: `http_plugin_old2.log`

__ d. Use the administrative console to start **webserver1**. Notice that a new `http_plugin.log` file is created in the logs directory.

__ 4. Clear the logs of server1.

__ a. Stop server1 from the administrative console. Click **Servers > Server Types > WebSphere application servers**.

__ b. Select **server1** and click **Stop**.

__ c. Wait for **server1** to stop.

__ d. Navigate to:

`/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1`

__ e. Delete all the log files. There are six log files in this directory (`native_stderr.log`, `native_stdout.log`, `startServer.log`, `stopServer.log`, `SystemErr.log`, `SystemOut.log`, `trace.log`, and `DumpNameSpace.log`).

__ 5. Start server1 from the administrative console.

__ a. Click **Servers > Server Types > WebSphere application servers**.

__ b. Select **server1** and click **Start**.

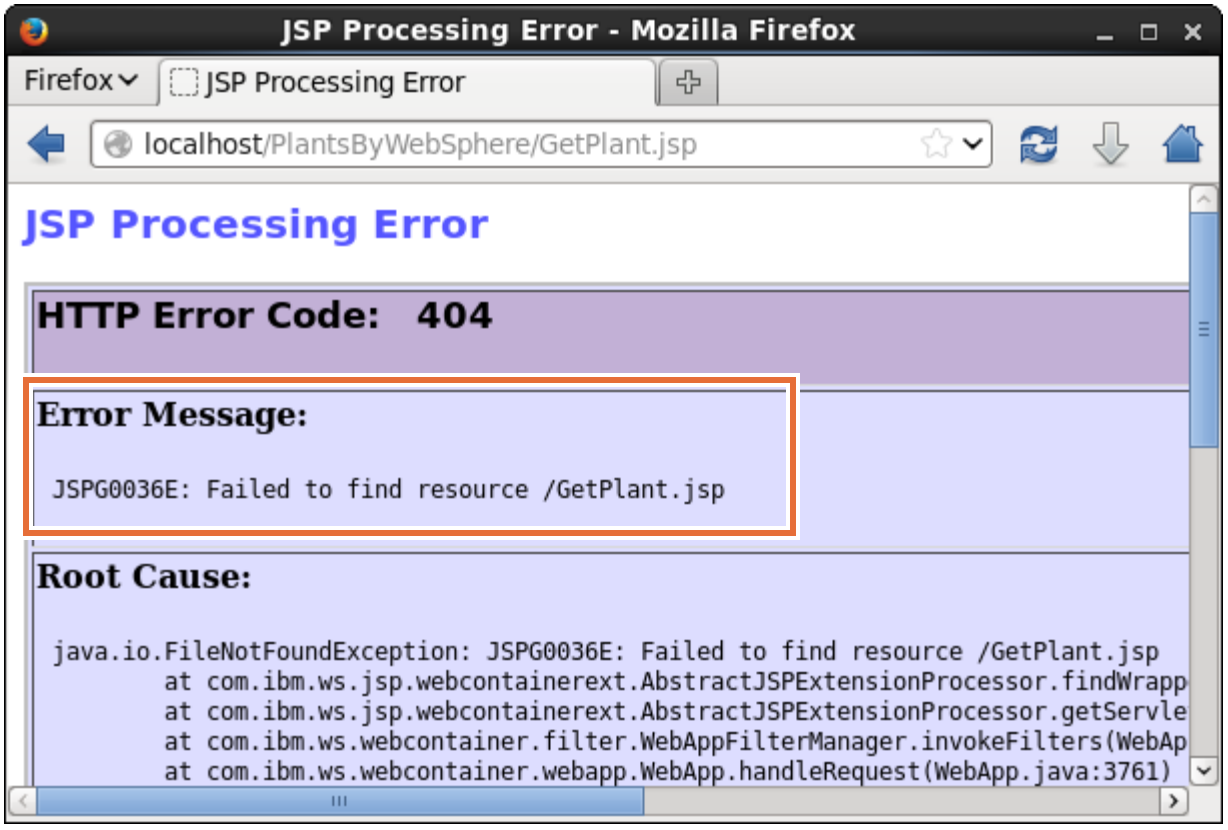
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Wait for **server1** to start.

__ 6. Submit a request for a web resource that the application serves.

__ a. Start a new browser window.

__ b. Type `http://localhost/PlantsByWebSphere/GetPlant.jsp` in the browser address bar.



__ c. Observe that the resource `GetPlant.jsp`, was not found.

__ d. Close the browser.

__ 7. Examine the HTTP server `access.log` file.

__ a. Navigate to `/opt/IBM/HTTPServer/logs`.

__ b. Open `/opt/IBM/HTTPServer/logs/access.log` file. You see a message like the following one:

```
127.0.0.1 - - [05/Feb/2014:10:17:22 -0500] "GET /PlantsByWebSphere/GetPlant.jsp
HTTP/1.1" 404 3093
```

__ c. Notice that the HTTP response code is 404, the File Not Found code.

__ d. Close the `access.log` file.

__ 8. Examine the `http_plugin.log` file that is at `/opt/IBM/WebSphere/Plugins/logs/webserver1`.

__ a. Open the `http_plugin.log` file.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Search for the following strings:

`webspherefindTransport`

and

`GetPlant.jsp`

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

and

htresponseRead

```
[05/Feb/2014:10:17:22.86073] 0000336f 69b9a710 - TRACE: ws_common:
websphereFindTransport: Finding the transport for server 0x1d1a310
[05/Feb/2014:10:17:22.86075] 0000336f 69b9a710 - TRACE: ws_common:
websphereFindTransport: Inbound is SSL:0 httptransports=1, https_transports=1
[05/Feb/2014:10:17:22.86077] 0000336f 69b9a710 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): washost on port 9080
[05/Feb/2014:10:17:22.86080] 0000336f 69b9a710 - DEBUG: ws_common:
websphereExecute: Executing the transaction with the app server reqInfo is 64009238
useExistingStream=0, client->stream=(nil)
[05/Feb/2014:10:17:22.86082] 0000336f 69b9a710 - DEBUG: ws_common:
websphereGetStream: Getting the stream to the app server (keepalive 28)
[05/Feb/2014:10:17:22.86084] 0000336f 69b9a710 - DEBUG: ws_transport:
transportStreamDequeue: Checking for existing stream from the queue
[05/Feb/2014:10:17:22.86088] 0000336f 69b9a710 - TRACE: ws_common:
websphereGetStream: Have a connect timeout of 5; Setting socket to not block for
the connect using FIONBIO
[05/Feb/2014:10:17:22.86298] 0000336f 69b9a710 - DEBUG: ws_common:
websphereGetStream: Setting socket to non-block for ServerIOTimeout over HTTP
[05/Feb/2014:10:17:22.86309] 0000336f 69b9a710 - DEBUG: ws_common:
websphereGetStream: socket 17 connected to washost:9080 timeout=900
[05/Feb/2014:10:17:22.86311] 0000336f 69b9a710 - DEBUG: lib_stream: openStream:
Opening the  stream soc=17
[05/Feb/2014:10:17:22.86317] 0000336f 69b9a710 - DETAIL: ws_common:
websphereGetStream: Created a new stream; queue was empty, socket = 17 Local
Port=37405
[05/Feb/2014:10:17:22.86320] 0000336f 69b9a710 - DEBUG: Set header
|_WS_HAPRT_WLMVERSION| to |-1|
[05/Feb/2014:10:17:22.86327] 0000336f 69b9a710 - DEBUG: lib_htrequest:
htrequestWrite: Writing the request reqInfo=64009238
URL=/PlantsByWebSphere/GetPlant.jsp size=0:
[05/Feb/2014:10:17:22.86329] 0000336f 69b9a710 - DETAIL:    GET
/PlantsByWebSphere/GetPlant.jsp HTTP/1.1
[05/Feb/2014:10:17:22.86331] 0000336f 69b9a710 - DETAIL:    Host: localhost
.
.
.
[05/Feb/2014:10:17:22.86354] 0000336f 69b9a710 - DEBUG: ws_common:
websphereExecute: Wrote the request; rea
ding the response
[05/Feb/2014:10:17:22.86357] 0000336f 69b9a710 - DETAIL: lib_htresponse:
htresponseRead: Reading the response: 64016a60
[05/Feb/2014:10:17:22.86359] 0000336f 69b9a710 - TRACE: lib_rio: Blocking for read,
waiting 900
[05/Feb/2014:10:17:23.30310] 0000338e 6db81700 - DEBUG: mod_was_ap22_http:
as_child_init pid= 0000338E
```


© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

```
[05/Feb/2014:10:17:23.46383] 0000336f 69b9a710 - DETAIL: HTTP/1.1 404 Not Found
```

Information

These lines indicate that the plug-in handles the request. The plug-in successfully sent the request to the application server, but the application server was unable to find the requested JSP page.

__ c. Close the `http_plugin.log` file.

__ 9. Examine the `trace.log` file of server1.

__ a. Navigate to:

`/opt/IBM/WebSphere/AppServer/profiles/profile1/logs/server1`

__ b. Open the `trace.log` file.

__ c. Search for: `JSPG0036E`. You find lines similar to the following messages:

```
2/5/14 10:17:22:978 EST] 00000090 SRTServletReq 1
com.ibm.ws.webcontainer.srt.SRTServletRequest setAttribu
te this->com.ibm.ws.webcontainer.srt.SRTServletRequest@59b790d0:  name -->
[jsp.resourceMissingErrorReport],  value -->
[com.ibm.ws.jsp.webcontainerext.JSPErrorReport: JSPG0036E: Failed to find resource
/GetPlant.jsp]
[2/5/14 10:17:22:978 EST] 00000090 SRTServletReq 1
com.ibm.ws.webcontainer.srt.SRTServletRequest attributeA
dded this->com.ibm.ws.webcontainer.srt.SRTServletRequest@59b790d0:  key -->
jsp.resourceMissingErrorReport value -->
com.ibm.ws.jsp.webcontainerext.JSPErrorReport: JSPG0036E: Failed to find resource
/GetPlant.jsp
[2/5/14 10:17:22:978 EST] 00000090 metadata      1
com.ibm.ws.webcontainer.metadata.WebModuleMetaDataImpl getJ2EEName()
J2EEname=PlantsByWebSphere#PlantsByWebSphereWeb.war,
mmd=com.ibm.ws.webcontainer.metadata.WebModuleMetaDataImpl@3e76e63c[PlantsByWebSph
ere#PlantsByWebSphereWeb.war]
```

Information

Notice that the plug-in log and the trace log both indicate that `GetPlant.jsp` cannot be found. Contact the application developers so that they can investigate why this resource cannot be accessed.

__ d. Close the `trace.log` file.

__ 10. Restore the default log level for the WebSphere HTTP server plug-in.

__ a. Click **Servers > Server Types > Web servers**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Click the link for **webserver1**.

__ c. Under **Additional Properties**, click **Plug-in properties.**

__ d. Scroll down and locate **Plug-in logging:**

__ e. Change the **Log level** to **Error**.

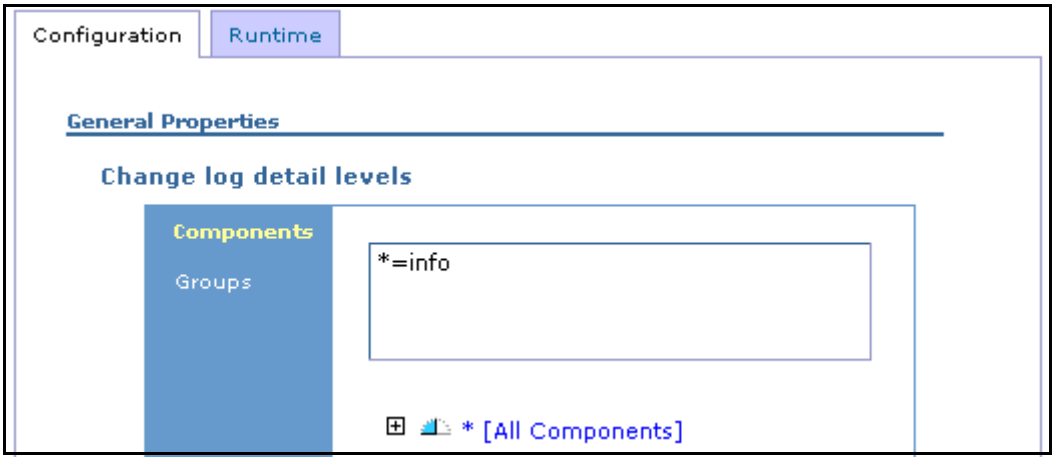
__ f. Click **OK**.

__ g. **Save** your changes to the master configuration. Wait for the synchronization process to complete. Click **OK**.

__ 11. Disable tracing on the web container of server1.

__ a. From the administrative console, select **Troubleshooting > Logs and Trace > server1 > Change Log Detail Levels**.

__ b. Edit the trace specification string to: `*=info`



__ c. Click **OK**, and **Save** the changes to the master configuration.

__ d. Wait for the nodes to synchronize and click **OK**.

__ e. Restart **server1**.

Section 6: Configuring a diagnostic provider on the web container

Diagnostic providers allow for a WebSphere Application Server component to be instrumented to export its own diagnostic provider object. A diagnostic provider object performs three functions: (1) dump the configuration of a component (2) dump the state of the component, and (3) run self tests on the component.

In this part of the exercise, you configure the web container diagnostic provider (WebContainerDP) to produce state information about the web container. This state dump includes information about cached servlets, virtual hosts, and general web container configuration information.

__ 1. Start server1 if it is not already started.

__ a. Click **Servers > Server Types > WebSphere application servers**.

__ b. Select **server1** and click **Start**.

__ c. Wait for **server1** to start.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ 2. Configure the web container diagnostic provider to produce a state dump from the web container.

__ a. Select **Troubleshooting > Diagnostic Provider > State Data**.

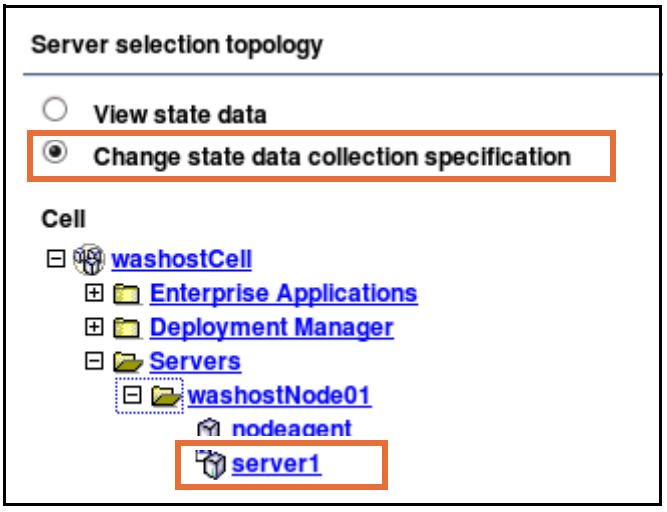
Note

When configuring a diagnostic provider, you can configure either state or configuration data. State data is used to check the health of a server runtime component. Configuration data is also used to check the health of a server runtime component. Runtime components with associated diagnostic configuration data include a diagnostic provider ID in their log entries.

Because two different types of data are collected, there are also two different memory dump types produced. A **configuration dump** is an operation that you can perform on a diagnostic provider to list the startup or current values of the configuration attributes for the DP. A **state dump** is similar to a configuration dump, but it differs in two key areas. First, a state dump displays current information about the operation of a component. The State Collection Specification values impact state dumps. State Collection Specification controls extra data collection that the component can perform at run time.

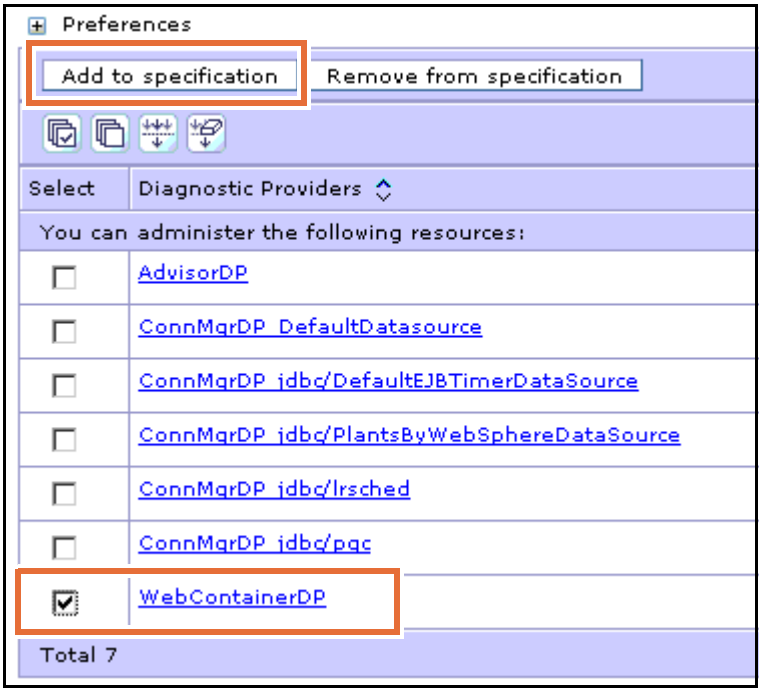
__ b. Select **Change state data collection specification**.

__ c. Expand **Cell**. Select **server1**.

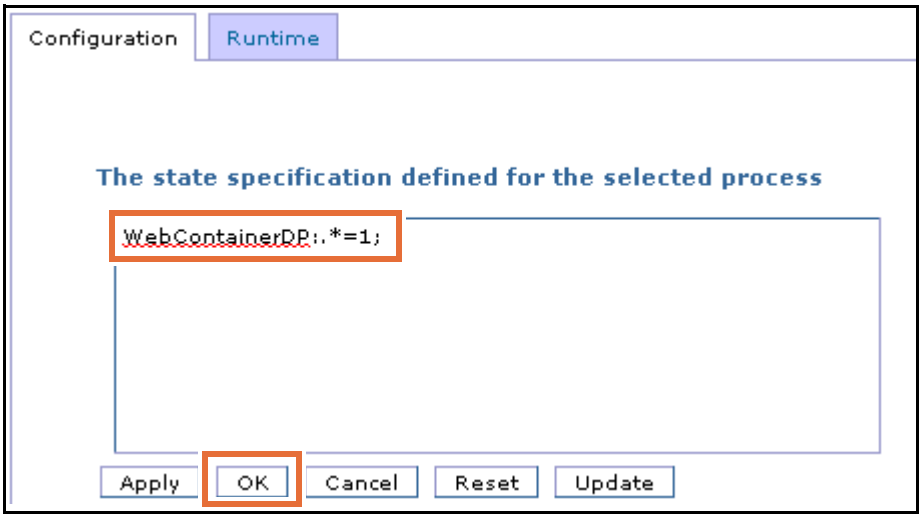


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. From the list of Diagnostic Providers, select **WebContainerDP**. Click **Add to specification**.



__ e. On the Configuration tab, the state specification was added. Click **OK**.



__ f. **Save** your changes to the master configuration.

__ g. Wait for your changes to synchronize; then click **OK**.

__ 3. Test the newly configured WebContainer Diagnostic Provider.

__ a. Start a new browser.

__ b. Type `http://localhost/PlantsByWebSphere` in the browser address bar.

© Copyright IBM Corp. 2013

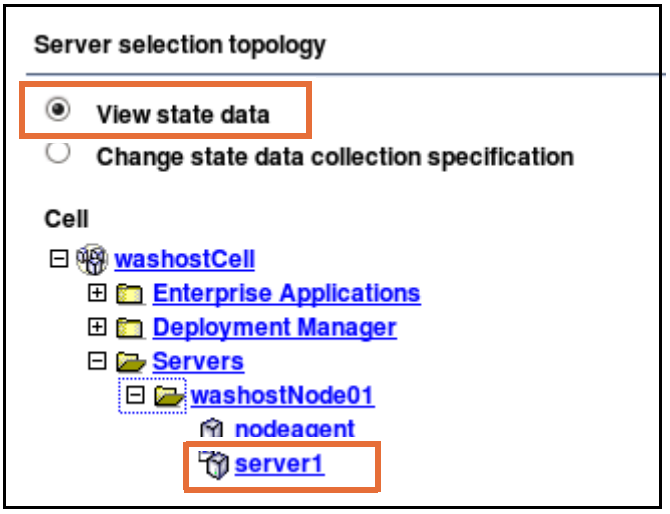
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. You see the **Plants By WebSphere** welcome page.

__ 4. View the state dump that the WebContainer Diagnostic Provider generated.

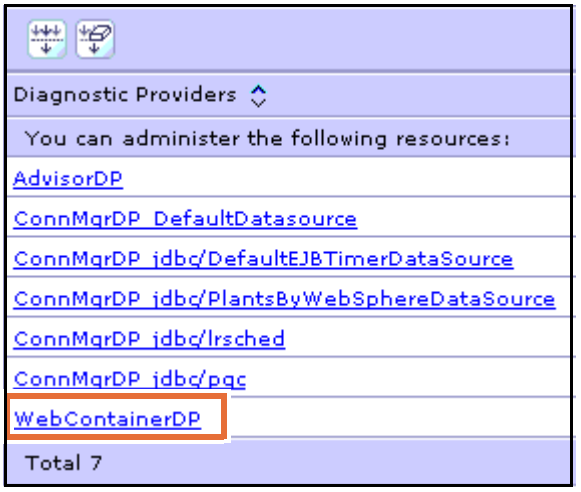
__ a. Select **Troubleshooting > Diagnostic Provider > State Data**.

__ b. Select **View state data.** Expand **Cell** and select **server1**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. From the list of Diagnostic Providers, click **WebContainerDP**.



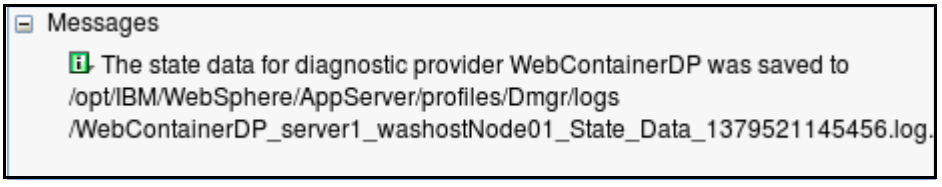
__ d. This action displays a large list of runtime state information from WebContainerDP. Scroll down the list until you find **PlantsByWebSphere** in the **Value** column. Examine the context root of the web module.

washostNode01	server1	cachedServlets-localhost:80/PlantsByWebSphere/javax.faces.resource/1x1_trans.gif.jsf-contextPath	/PlantsByWebSphere	The context root of the webmodule this cached servlet wrapper belongs to.

__ 5. Save the state dump to a log file.

__ a. Scroll up, and click **Save**.

__ b. A message indicates the location of the log file.



__ c. Navigate to the location of the state data log file.

__ d. Open the file. Scroll through the state data log file. Close it when finished.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Section 7: Use IBM Web Server Plug-in Analyzer for WebSphere Application Server [Desktop] to analyze plug-in data

In this section, you use the plug-in analyzer tool to examine the WebSphere topology. In addition you can use the plug-in analyzer tool to look at the errors you saw in the logs previously.

__ 1. Make sure that IBM Support Assistant is started. You started it in the previous sections, but if you stopped it, follow these steps:

__ a. Start the IBM Support Assistant by clicking **Applications** on the toolbar. Select **Start IBM Support Assistant 5**.

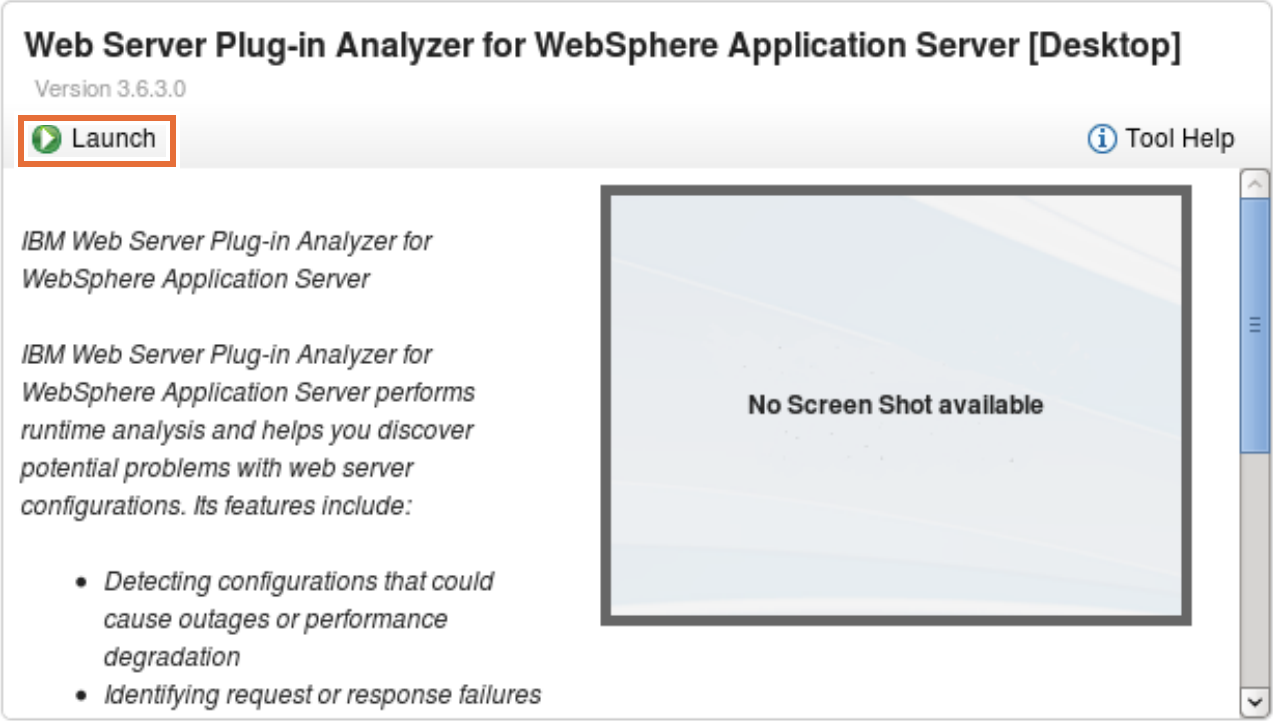
__ b. A terminal window pops up. Wait until you see a message that says the IBM Support Assistant is started. Do not close this window until you are finished with the IBM Support Assistant.

__ 2. Launch the **IBM Web Server Plug-in Analyzer for WebSphere Application Server [Desktop]** from the **IBM Support Assistant**.

__ a. Open a web browser, and enter the web address:

http://washost:10911/isa5

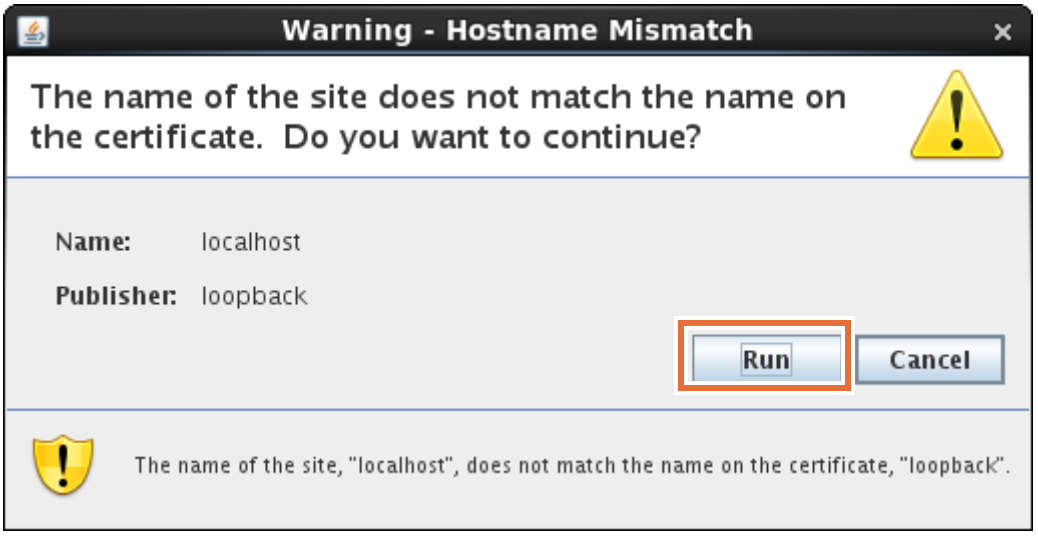
__ b. Launch the **Plug-in Analyzer [Desktop]** from the **Tools** tab of the IBM support Assistant. Click on **Launch**.



__ c. Click **Submit**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Click **Run** if prompted about a hostname mismatch.



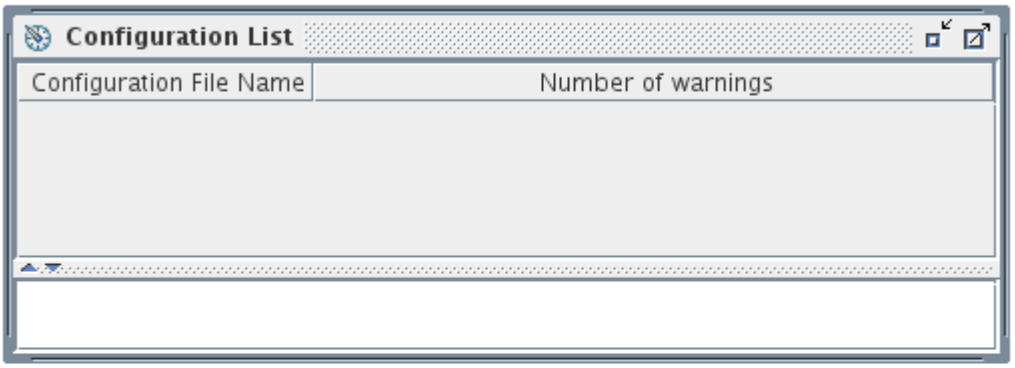
© Copyright IBM Corp. 2013

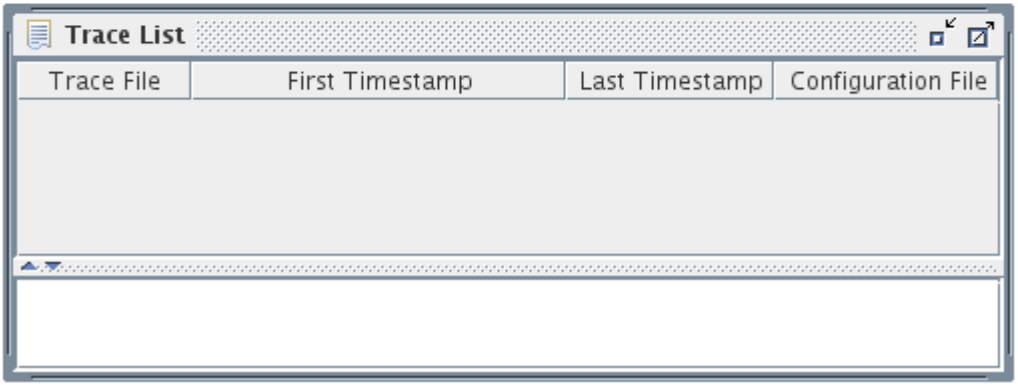
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Wait for the Java Web Start to download the application and for the tool to start.

__ 1. look at the plug-in analyzer and load the files to analyze.

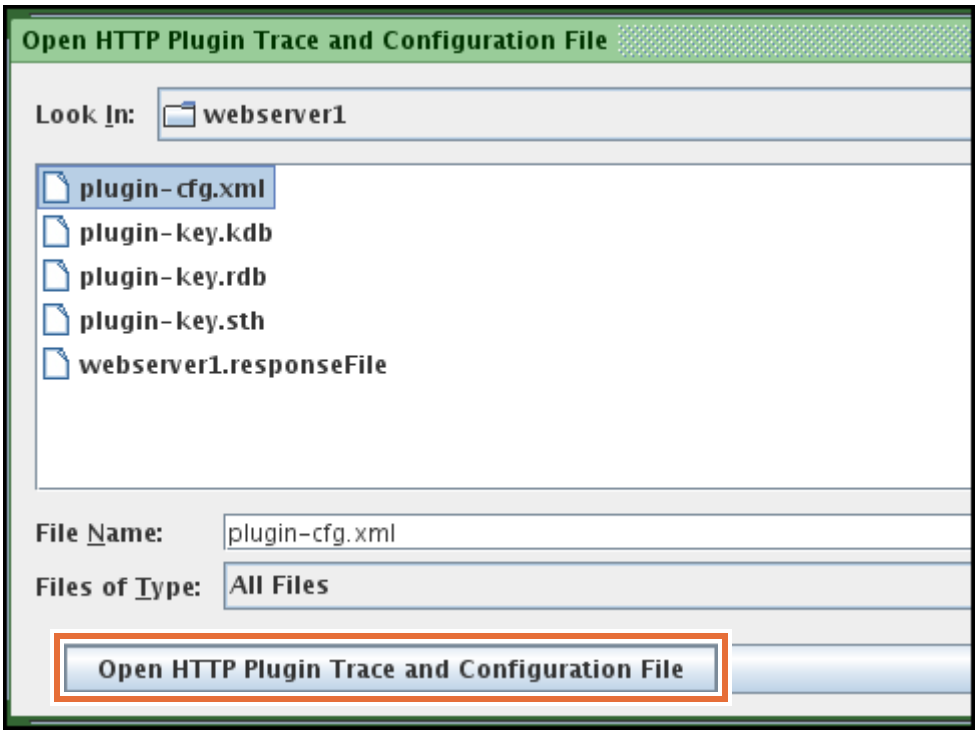
__ a. At the top, there are two windows, **Configuration List** and **Trace List**.



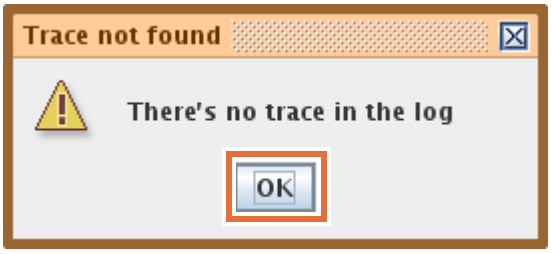


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. From the **File** menu, select **Open Trace and Configuration File** item. Navigate to **/opt/IBM/WebSphere/Plugins/config/webserver1** and select the file **plugin-cfg.xml**. Click **Open HTTP Plugin Trace and Configuration File**.



__ c. Click OK if you receive a message the "**There's no trace in the log**"



__ d. From the **File** menu, select **Open Trace and Configuration File** item. Navigate to **/opt/IBM/WebSphere/Plugins/logs/webserver1** select the **http_plugin_old.log** file. Click **Open HTTP Plugin Trace and Configuration File**.

__ e. Repeat this process for the **http_plugin_old2.log** file.

__ 2. Use the plug-in analyzer tool to review the plug-in configuration file.

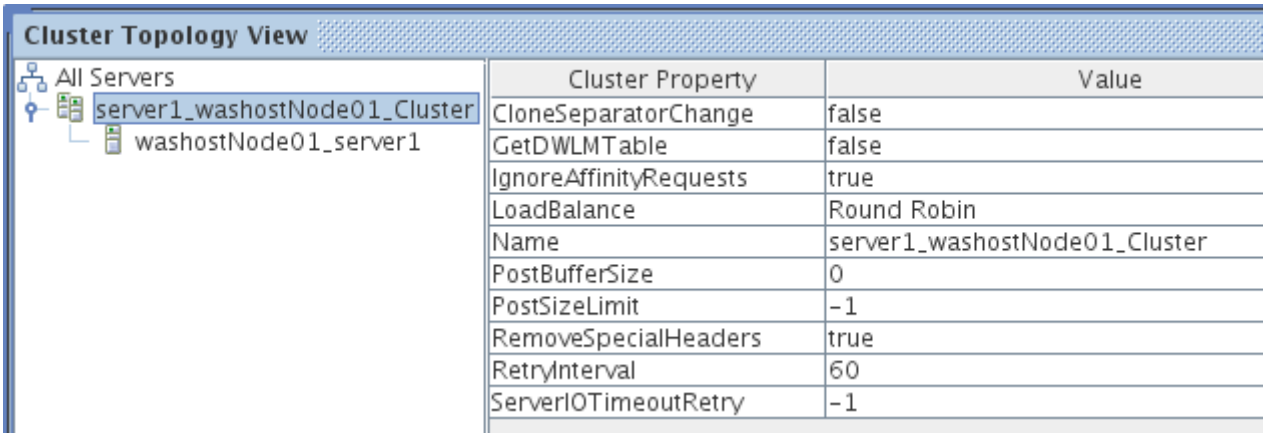
__ a. From the **Analysis** menu, select the **Cluster Topology View**.

__ b. Double-click on **All Servers**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Click on **server1_washostNode01_Cluster**.



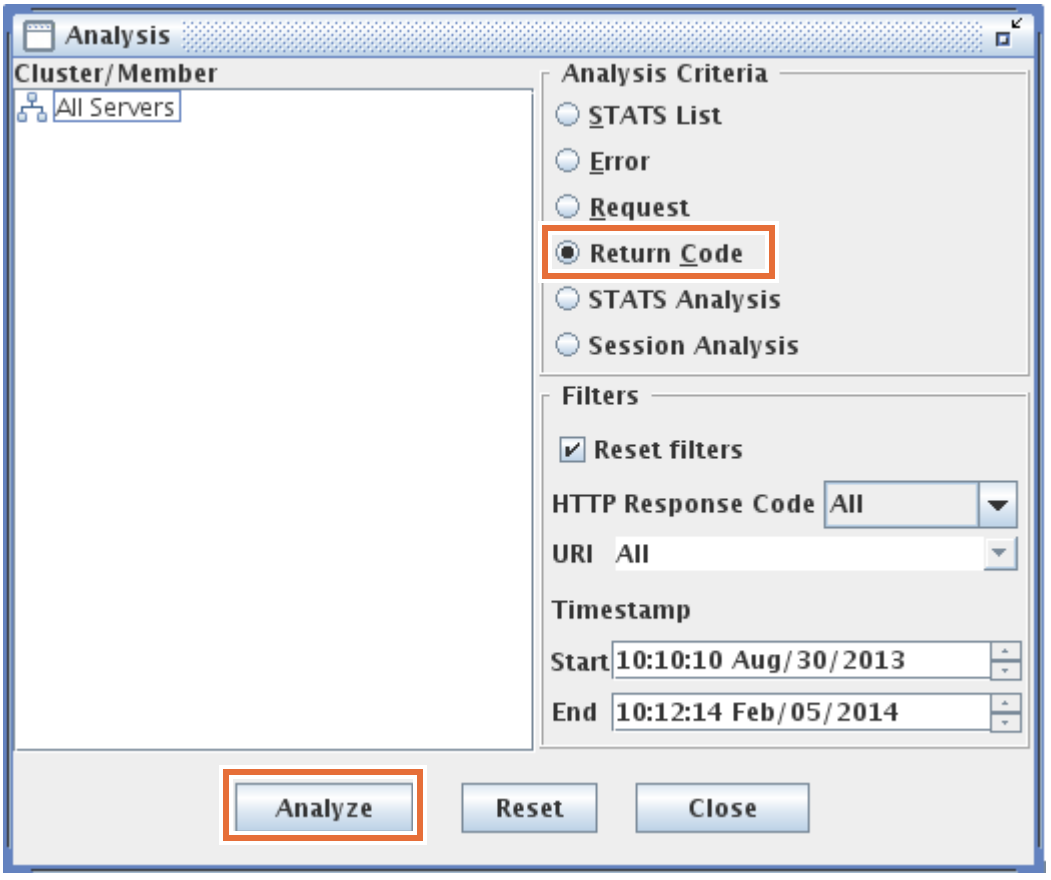
__ d. Take time to review the information presented in the diagram and the table. Click to icon next to the cluster name to see the servers that are contained in this cluster.

__ e. Close the **Cluster Topology View** when you are done.

__ 3. Use the plug-in analyzer tool to examine the plug-in logs.

__ a. From the **Analysis** menu select the **Plug-in Trace Analysis**.

__ b. Select **Return Code** as the **Analysis Criteria** and then click **Analyze**.



__ c. Look at the Return Code Analysis window. Can you find the error that you saw in the section where you examined the results of a failed web request?

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Previously, you used index99.html as the URI to generate an error.

Return Code Analysis

Time(ms)	Return Code	URL	Trace	Line Nu
-90,169	No route found	/index.html	[05/Feb/2014:0...	358
18	No route found	/images/administration.gif	[05/Feb/2014:0...	721
19	No route found	/favicon.ico	[05/Feb/2014:0...	1228
20	No route found	/http_server_styles.css	[05/Feb/2014:0...	858
20	No route found	/images/foreground.gif	[05/Feb/2014:0...	910
21	No route found	/images/favicon.ico	[05/Feb/2014:0...	1172
22	No route found	/images/support.gif	[05/Feb/2014:0...	875
23	No route found	/images/odot.gif	[05/Feb/2014:0...	705
23	No route found	/favicon.ico	[05/Feb/2014:0...	1210
25	No route found	/images/background.gif	[05/Feb/2014:0...	894
26	No route found	/index99.html	[05/Feb/2014:0...	1191
661	0	/PlantsByWebSphere	[05/Feb/2014:0...	303
14,800	No route found	/images/notes.gif	[05/Feb/2014:0...	855
15,842	No route found	/images/help.gif	[05/Feb/2014:0...	738

__ e. Click on the index99.html URL and examine the lower left part of the window.

Request Summary

Request began = 05/Feb/2014:09:27:26.40743
Hostname = localhost
URI = /index99.html
VirtualHostGroup = default_host
[05/Feb/2014:09:27:26.40769] 0000301c 85baf710 -
DETAIL: ws_common: websphereShouldHandleRequest:
No route found

__ f. On the lower right side is the detail of the log file. You can click on a summary message on the left and the log detail will go to the corresponding entry.

Request Summary

Request began = 05/Feb/2014:09
Hostname = localhost
URI = /index99.html
VirtualHostGroup = default_host
[05/Feb/2014:09:27:26.40769] 00

Request Trace

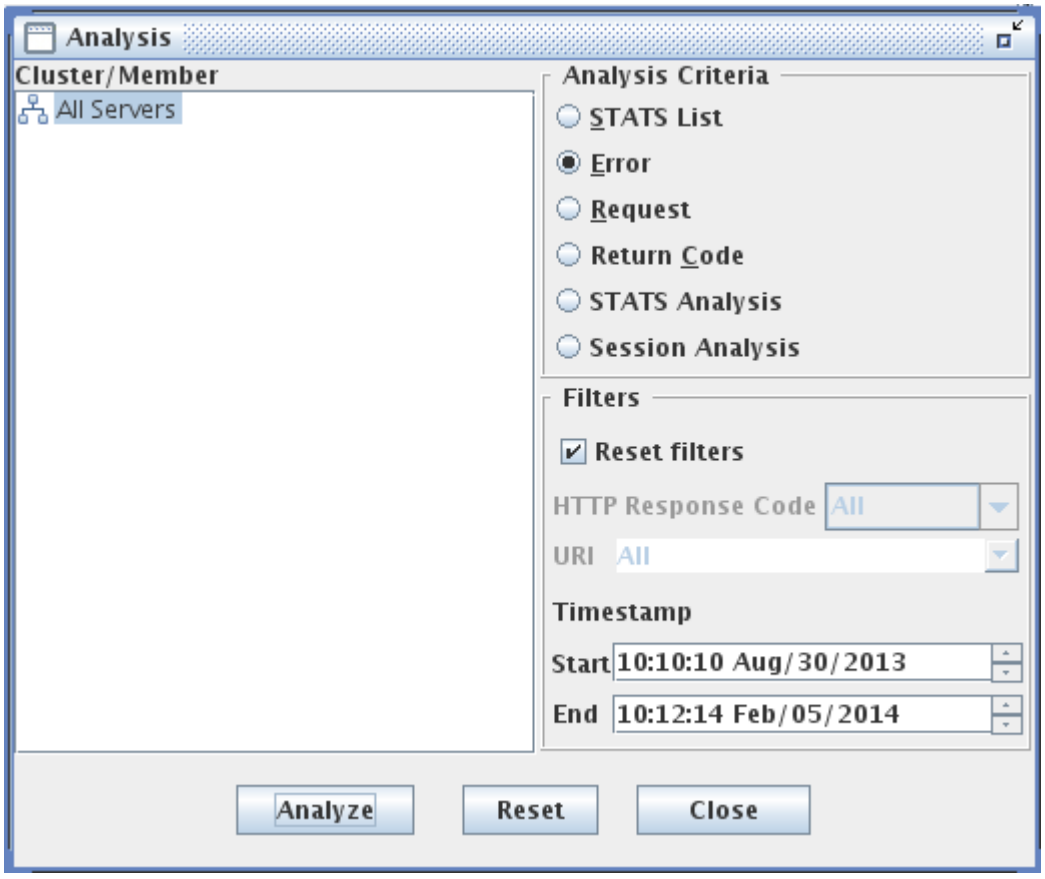
[05/Feb/2014:09:27:26.40743] 0000301c 85baf710 - DEBUG: lib_util: parseHostHeader: Host: 'localhost', port 80
[05/Feb/2014:09:27:26.40744] 0000301c 85baf710 - DEBUG: ws_common: websphereCheckConfig: Current time is 1391610446, next stat time is 1391610210
[05/Feb/2014:09:27:26.40748] 0000301c 85baf710 - DEBUG: ws_common: websphereCheckConfig: Latest config time is 1391609332, lastModTime is 1391609332
[05/Feb/2014:09:27:26.40750] 0000301c 85baf710 - DETAIL: ws_common: websphereShouldHandleRequest: trying to match a route for: vhost='localhost'; uri='/index99.html'
[05/Feb/2014:09:27:26.40753] 0000301c 85baf710 - DEBUG: ws_common: websphereShouldHandleRequest: NOT config->odrEnabled(reqInfo(68006e18))
[05/Feb/2014:09:27:26.40754] 0000301c 85baf710 - TRACE: ws_common: websphere PortNumberForMatching: Using logical.
[05/Feb/2014:09:27:26.40756] 0000301c 85baf710 - TRACE:

__ g. Take time to examine the Return Code Analysis and close it when you are finished.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ h. Use the Error Analysis Criteria to examine the log. Can you find the error you saw previously when the server was not ready?



End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In these exercises, you followed requests for various web resources, including HTML files and JSPs. You verified how each component in the request flow handled the request by examining log files. You enhanced the information that was written to these log files by configuring tracing on different components such as the WebSphere HTTP server plug-in and the application server. Finally, you learned how to use two tools for gathering diagnostic data about the web container and the HTTP plug-in.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 12.Troubleshooting WebSphere default messaging

What this exercise is about

This exercise demonstrates techniques for debugging message flow problems in WebSphere Default Messaging.

What you should be able to do

At the end of the lab, you should be able to:

- Identify default messaging symptoms
- Examine log files for messaging-related errors
- Configure the tracing of messaging components
- Analyze trace logs to determine root causes
- Use JMS clients to verify messaging functions
- Use the IBM Service Integration Bus Destination Handler

Introduction

This exercise is meant to be a brief introduction to the WebSphere Default Messaging component of WebSphere Application Server. You build a bus, and add members. Then, you produce and consume messages by using the IBM Service Integration Bus Destination Handler.

Required materials

All of the files necessary to complete this exercise are stored on a special directory on your lab VMware image. The IBM Service Integration Bus Destination Handler must be preinstalled on the lab image.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Start the deployment manager and create a second application server

__ 1. Start the deployment manager.

__ a. From a command prompt change directory to `<profiles_root>/Dmgr/bin`

__ b. Enter the command: `./startManager.sh`

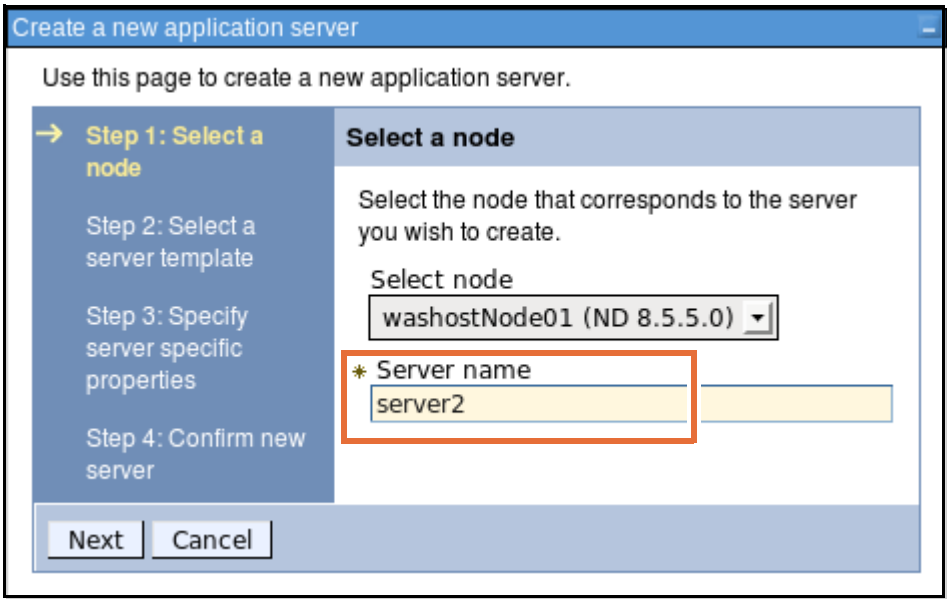
__ 2. Create another application server called server2.

__ a. Log on to the administrative console with `wasadmin` and `web1sphere`

__ b. Click **Servers > Server Types > WebSphere application servers**.

__ c. Click **New**.

__ d. In the Server name field, enter `server2` and click **Next**.



__ e. Accept the **default** server template in **Step 2** and click **Next**.

__ f. Make sure that **Generate Unique Ports** is selected in **Step 3** and click **Next**.

__ g. Read the summary in **Step 4** click **Finish**.

__ h. Click **Save** to save directly to the master configuration.

__ i. Wait for the cell update to complete and click **OK**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 2: Create a bus and add two application servers to the bus as members

__ 1. Start the node agent if it is not already started.

__ a. From a command prompt change directory to: `<profiles_root>/profile1/bin`

__ b. Enter the command: `./startNode.sh`

__ c. Wait for the node agent to start.

__ 2. Start the application servers.

__ a. From the administrative console click **Servers > Server Type > WebSphere application servers**.

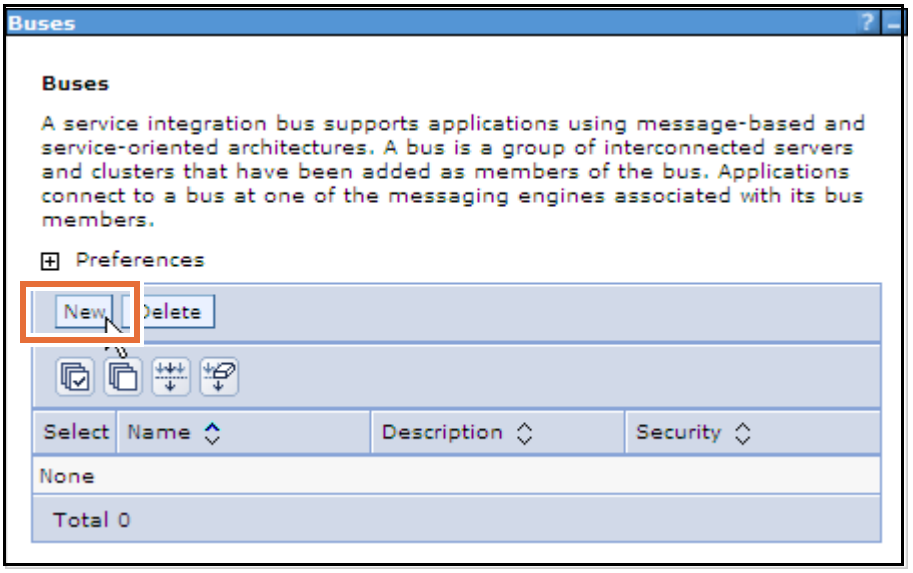
__ b. Check the boxes next to **server1** and **server2**, and then click **Start**.

__ c. Wait for both servers to start.

__ 3. Create a service integration bus.

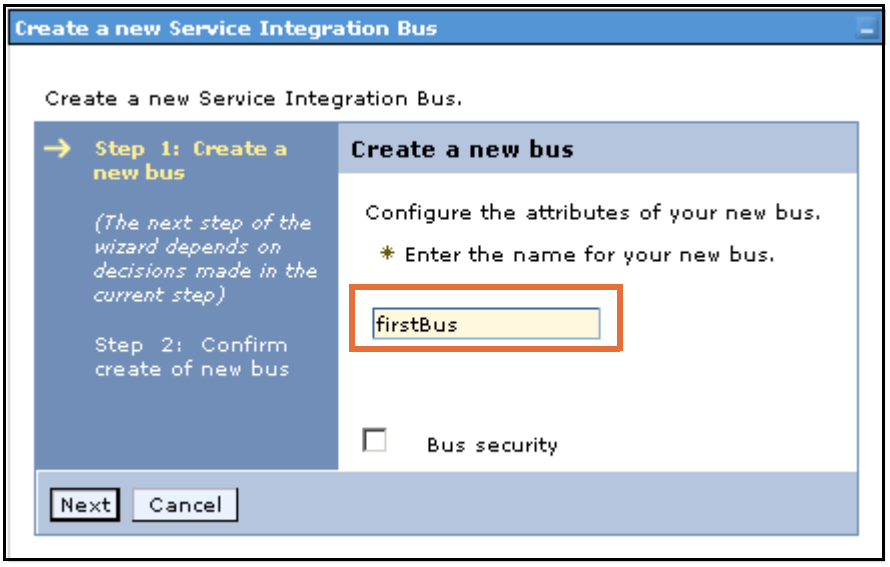
__ a. Click **Service integration > Buses**.

__ b. Click **New**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Enter `firstBus` for the bus name and clear the box for **Bus security**.



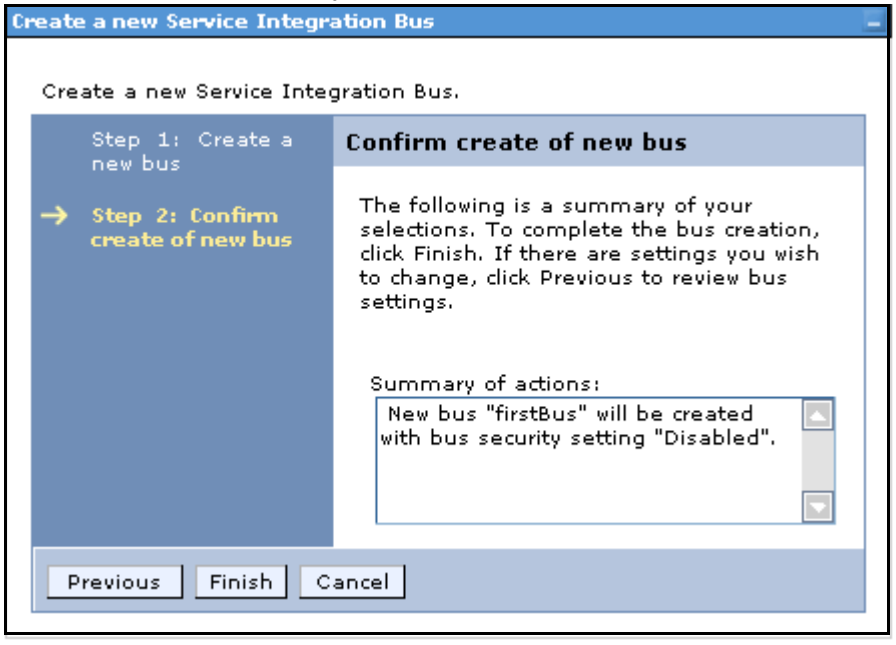
Note

Bus security:

Since this exercise does not use bus security, not enabling it saves some time. In practice, it is suggested that you enable bus security.

__ d. Click **Next**.

__ e. On **Step 2**, read the summary and click **Finish**.



__ f. Click **Save** to save directly to the master configuration.

© Copyright IBM Corp. 2013

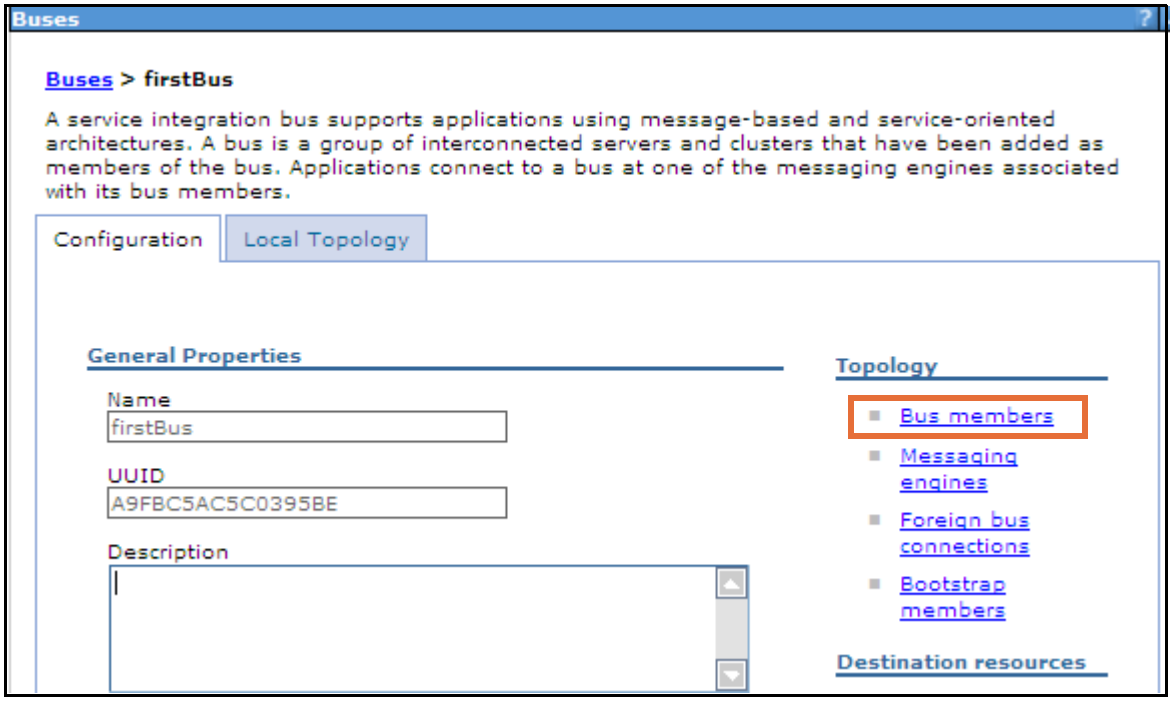
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ g. Wait for the nodes to synchronize and click **OK**.

__ 4. Add server1 as a member of firstBus.

__ a. Click **firstBus**.

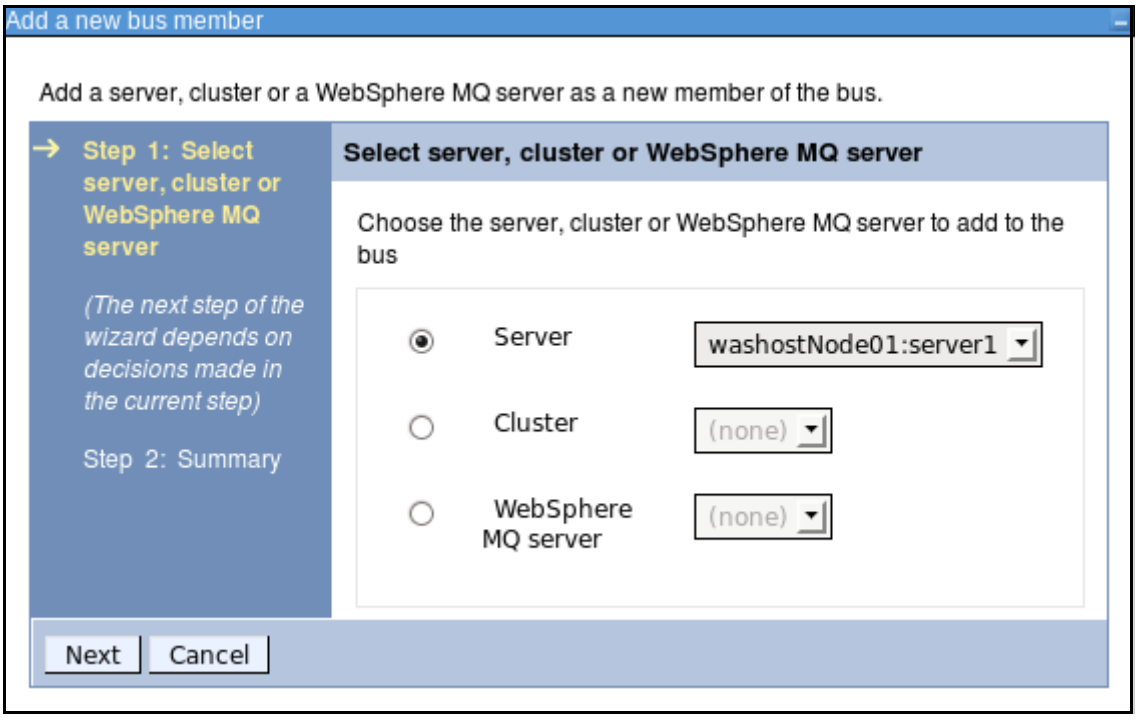
__ b. Click **Bus members**.



__ c. Click **Add**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. The server **washostNode01:server1** is selected by default; click **Next** to continue.



__ e. Notice that **File store** is selected by default. Click **Next** to continue.

__ f. Accept the defaults on the **file store properties** panel and click **Next** to continue.

__ g. Accept the defaults on the **Tune performance parameters** panel and click **Next**.

__ h. Click **Finish** on the summary panel to add server1 as a member of firstBus.

__ i. Click **Save** to save directly to the master configuration.

__ j. Wait for the nodes to synchronize and click **OK**.

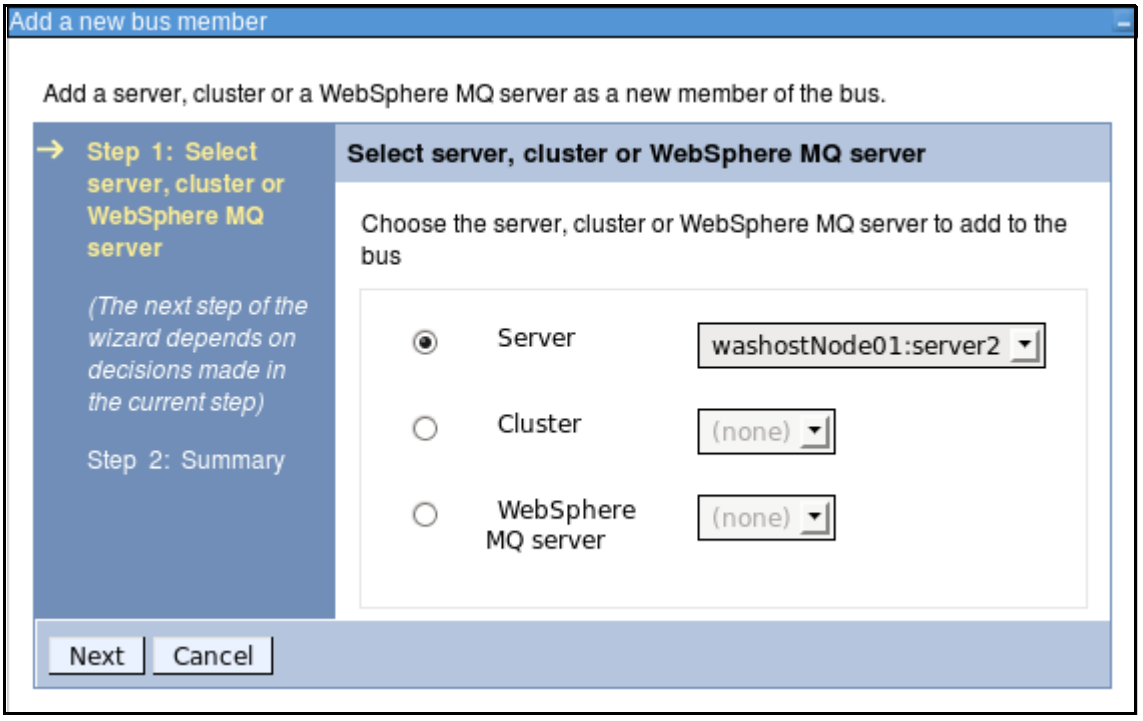
__ 5. Add server2 as a member of firstBus.

__ a. Click **Add**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. The server **washostNode01:server2** is selected by default; click **Next** to continue.



__ c. **File store** is selected by default. Click **Next**.

__ d. Accept the defaults on the **file store properties** panel and click **Next**.

__ e. Accept the defaults on the **Tune performance parameters** panel and click **Next**.

__ f. Click **Finish** on the summary panel to add server2 as a member of firstBus.

__ g. Click **Save** to save directly to the master configuration.

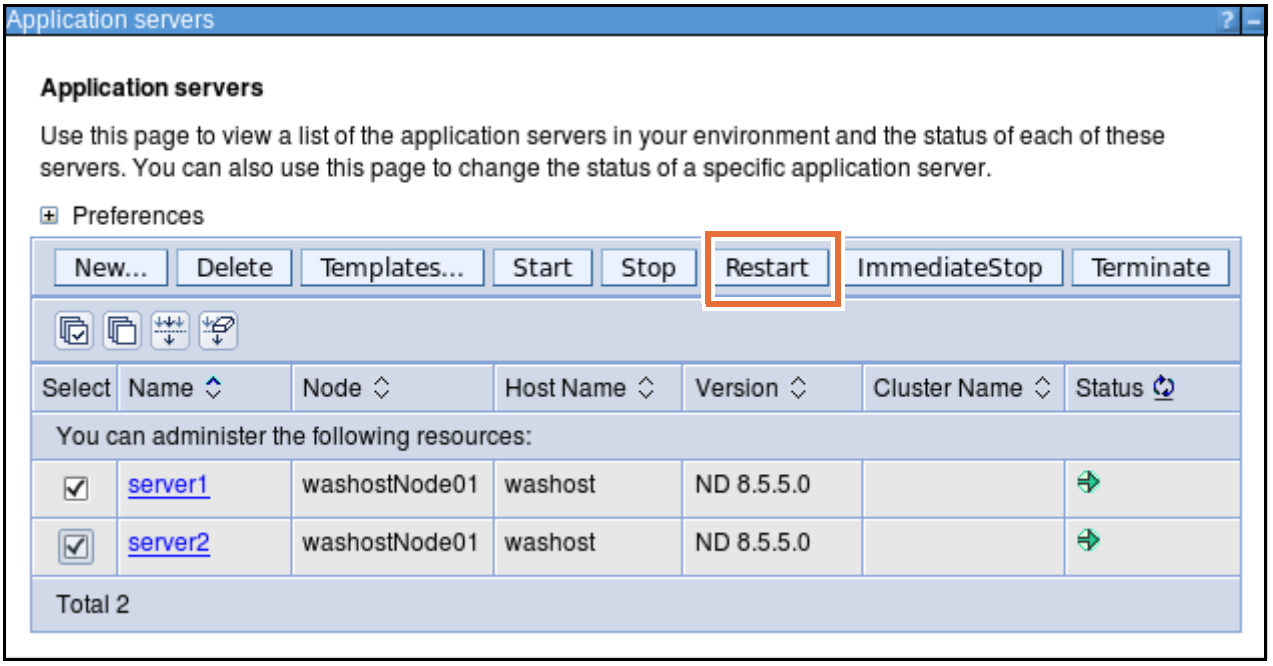
__ h. Wait for the nodes to synchronize and click **OK**.

__ 6. Restart the application servers. The servers need to be restarted so that they can read the configuration for their message engines and start them.

__ a. From the administrative console click **Servers > Server Types > WebSphere application servers**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Check the boxes next to both **server1** and **server 2** and click **Restart**.



__ c. Wait for each server to start successfully.

__ d. Check the `SystemOut.log` file for each server. You see messages like the following, indicating that the message engine started:

```
. . .
CWSID0016I: Messaging engine washostNode01.server2-firstBus is in state
Started
```

```
00000066 SibMessage    I    [firstBus:washostNode01.server2-firstBus]
CWSID0016I: Messaging engine washostNode01.server2-firstBus is in state Started
```

Section 3: Create a bus destination

__ 1. Verify that the message engines are running.

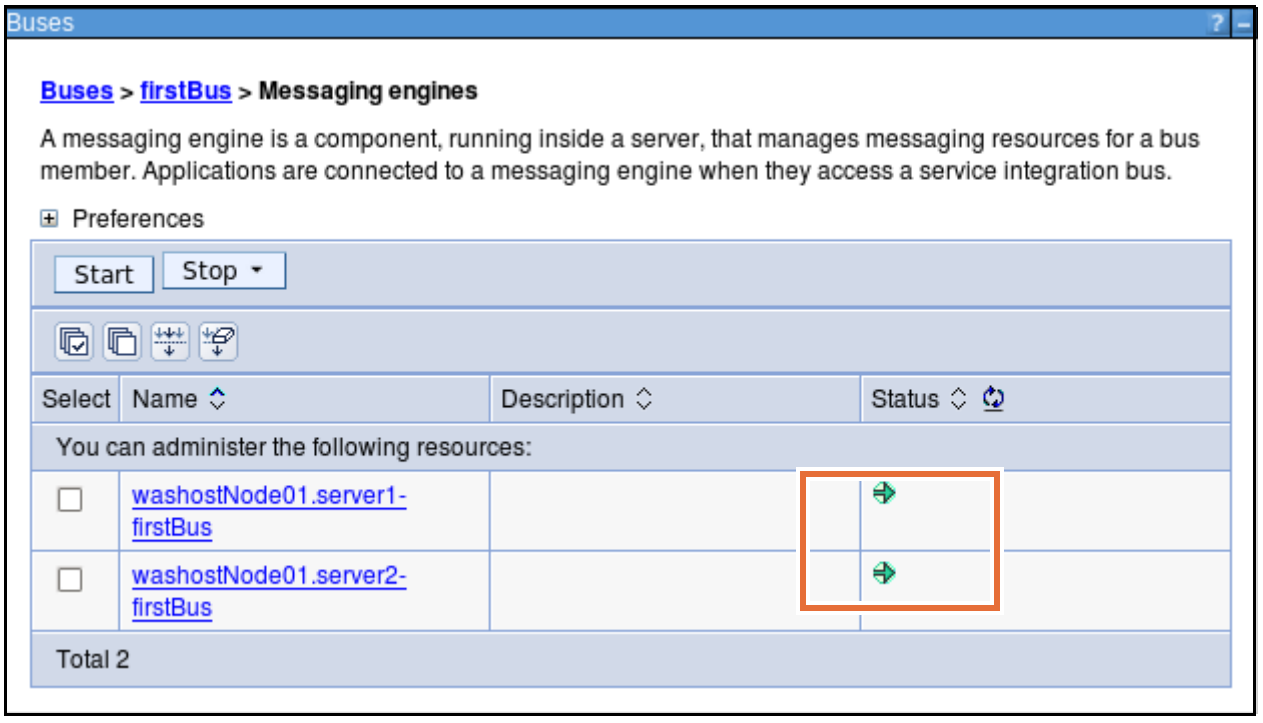
__ a. From the administrative console, click **Service integration > Buses > firstBus > Messaging engines**.

__ b. Notice that the names of the messaging engines are in the form of `NodeName.ServerName-Busname`.

© Copyright IBM Corp. 2013

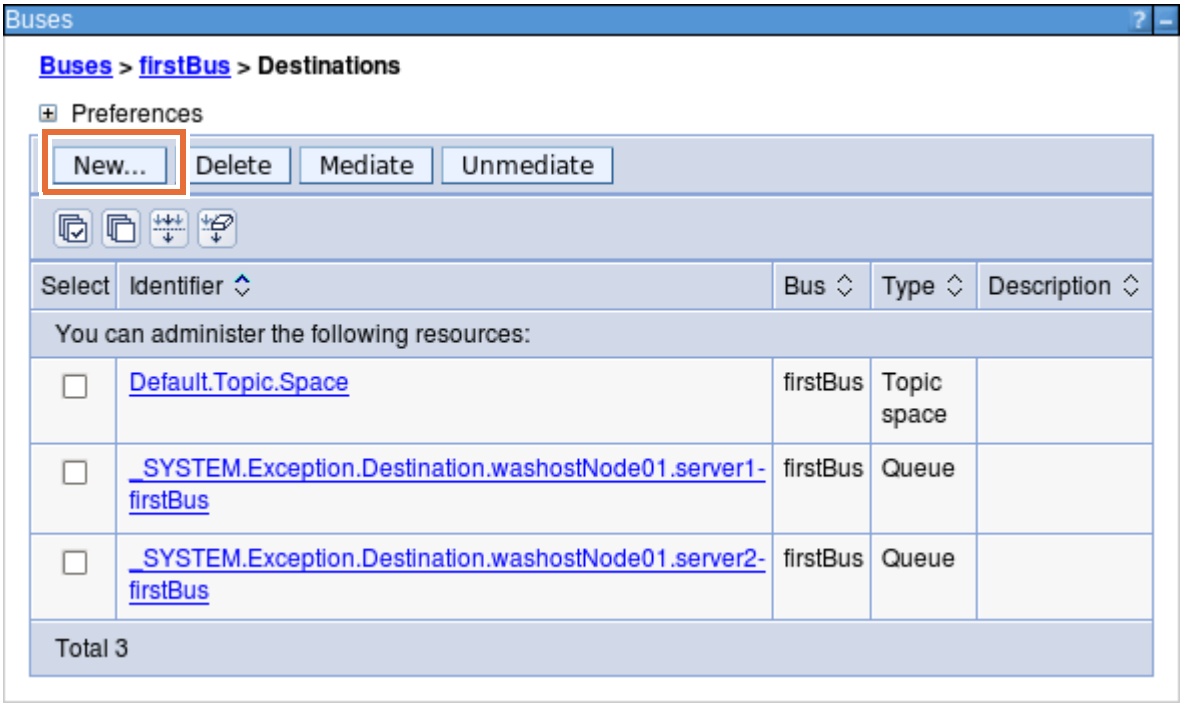
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. There are green arrows indicate that the messaging engines are in a started state. The bus is ready to conduct messaging business.



__ 2. Create a queue destination.

__ a. From the administrative console, click **Service integration > Buses > firstBus > Destinations**. Notice the three destinations that are created by default. One is the default topic space, and the others are the default exception destinations for each messaging engine.

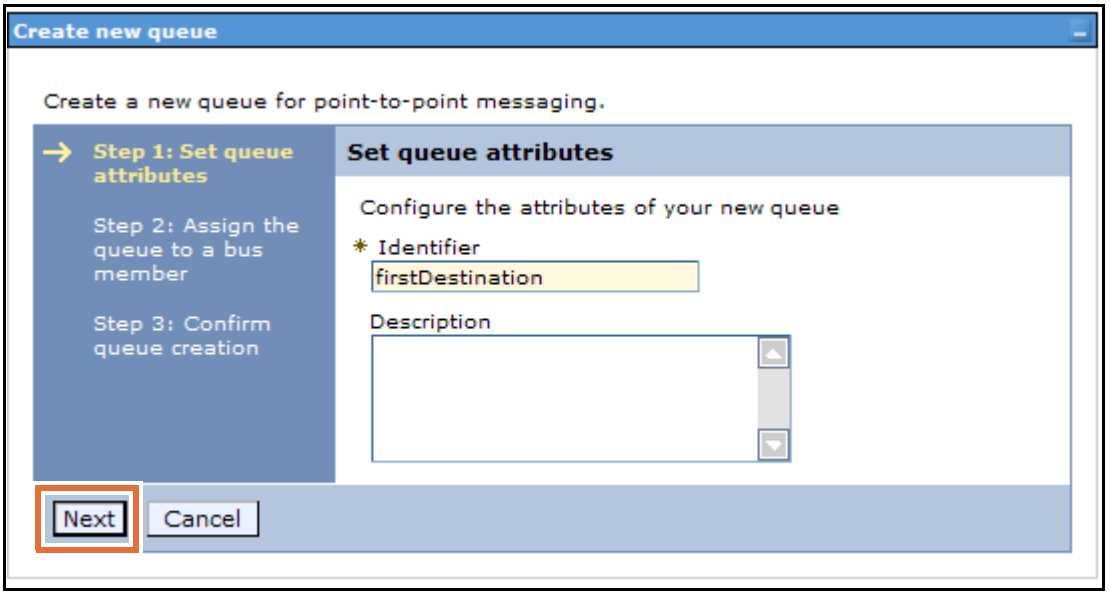


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

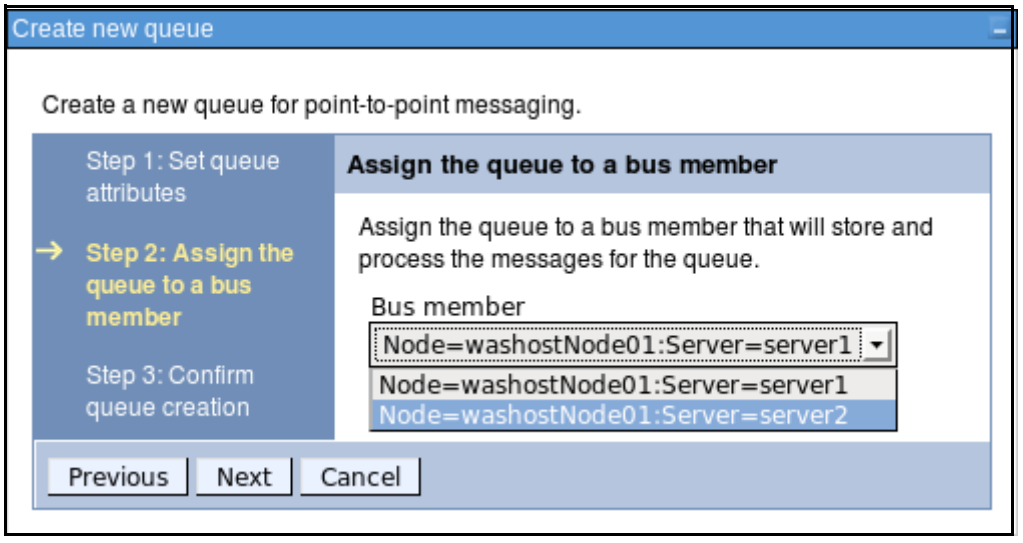
__ b. Click **New** to create a destination.

__ c. Accept the default **Queue** for destination type, and click **Next**.

__ d. On the **Set queue attributes** panel, complete the identifier field; call the queue `firstDestination` and click **Next**.



__ e. On the **Assign the queue to a bus member** panel, use the menu to select the messaging engine on **server2** and click **Next**.



Note

You are making the destination local to the messaging engine on **server2**. Local means that the actual **queue point** of this destination lives in the messaging engine on server2.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. On the **Confirm queue creation** panel, select **Finish** to create the queue.

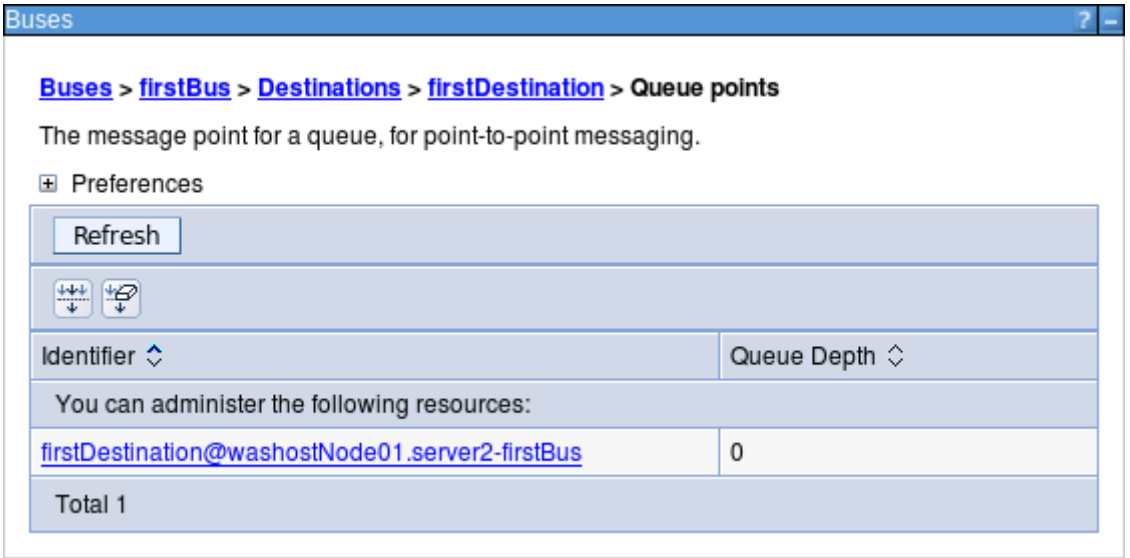
__ g. Click **Save** to save directly to the master configuration.

__ h. Wait for the nodes to synchronize and click **OK**.

__ 3. Use the administrative console to view the queue point for firstDestination.

__ a. Click **Service integration > Buses > firstBus > Destinations > firstDestination > Queue points**.

__ b. Notice the location of the queue point. The queue point for firstDestination is local to the messaging engine called **washostNode01.server2-firstBus**, the messaging engine on server2.



Section 4: Connect to the bus with the IBM Service Integration Bus Destination Handler

The IBM Service Integration Bus Destination Handler is preinstalled on your lab images. It is a tool for the default messaging provider for WebSphere Application Server V6.0, V6.1, V7.0, V8.0, and V8.5.

The tool can perform various actions on messages in a service integration bus without writing any custom code.

These actions can be run one time to complete an individual task, or the tool can be deployed as a WebSphere Application Server scheduler task. It can regularly check the contents of an exception destination and provide appropriate handling of messages that are based on the set of rules configured.

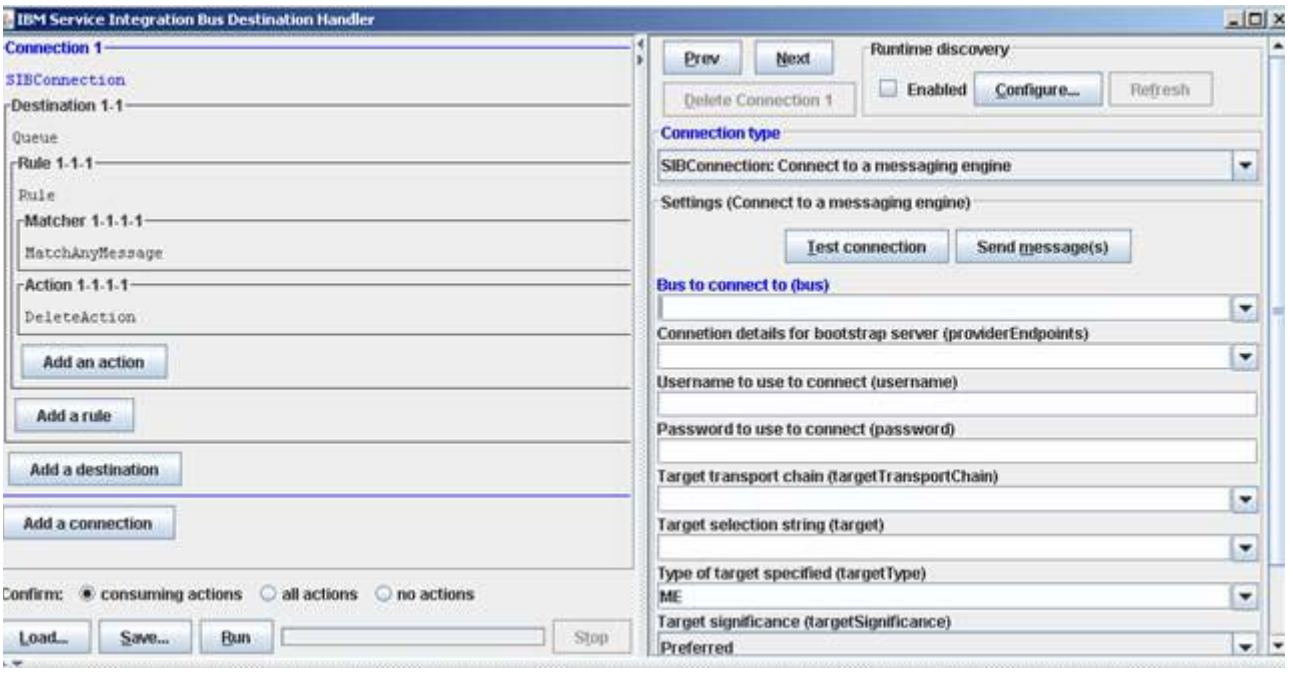
__ 1. Start the IBM Service Integration Bus Destination Handler.

__ a. Open a command prompt and change directory to: `<profiles_root>/profile1/bin`.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Enter the following command on one line (or copy and paste it from `/usr/Student/DefaultMessaging/SIBClientcmd.txt`):

```
./launchClient.sh sib.desthandler.appclient.ear –CCBootstrapHost=washost
–CCBootstrapPort=9810
```



Information

The **IBM Service Integration Bus Destination Handler** can aid with the following tasks.

It can perform individual actions in an interactive way, without needing a configuration file. For example:

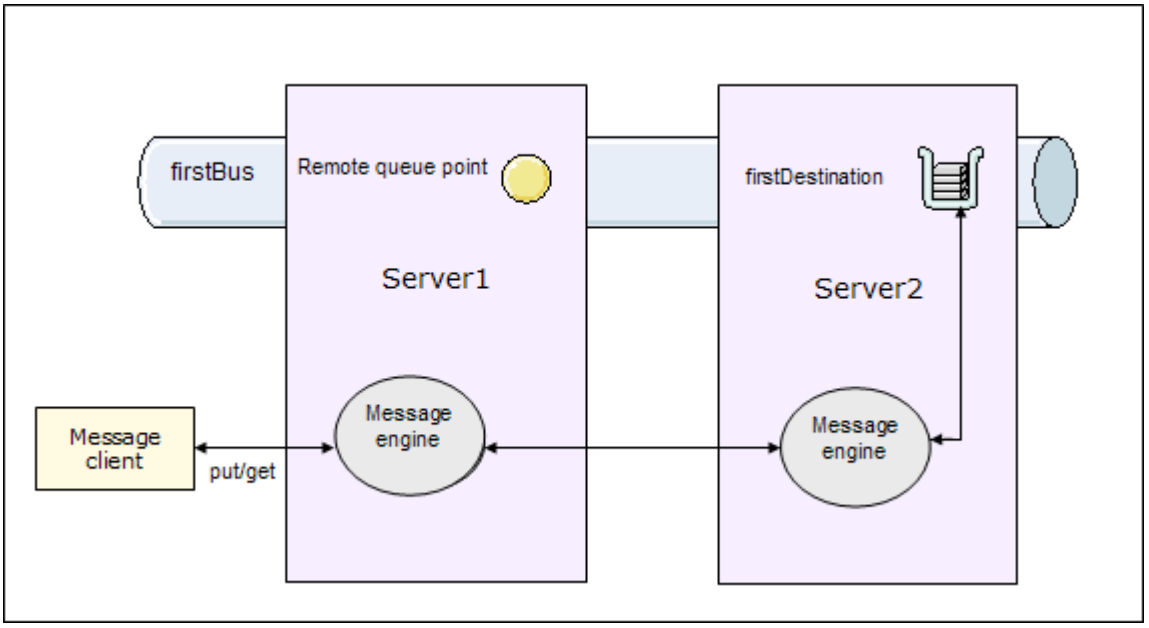
- Viewing the contents of a queue, and saving the output to a file
- Moving messages from one destination to another, with a confirmation dialog for each message
- Sending messages into the bus to test rules
- Building and testing a configuration file for running the tool on the command line, or as a scheduled task
- Exploring the runtime environment to determine connection and destination information

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

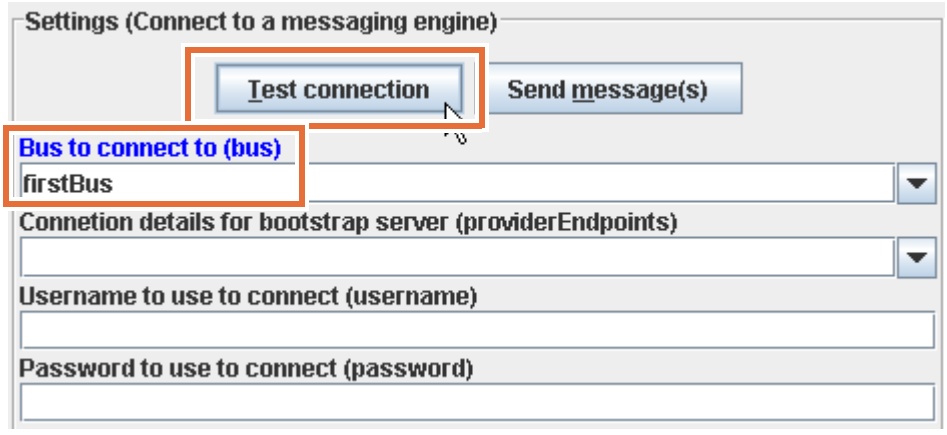
Section 5: Produce a message by using the IBM Service Integration Bus Destination Handler

In this part of the exercise, you use the messaging client IBM Service Integration Bus Destination Handler to put a message on the queue firstDestination. Then, in the next part, you consume the message.



__ 1. Put a message on the firstDestination queue.

__ a. In the **Bus to connect to(bus)** field, enter: `firstBus`

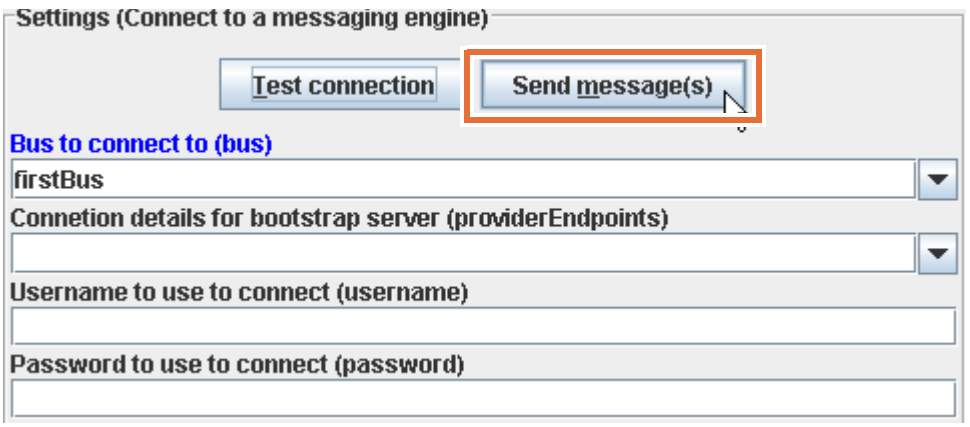


__ b. Click **Test connection**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. You see a box that confirms that the connection was successful. Click **OK** to close it.

__ d. Click **Send message**.



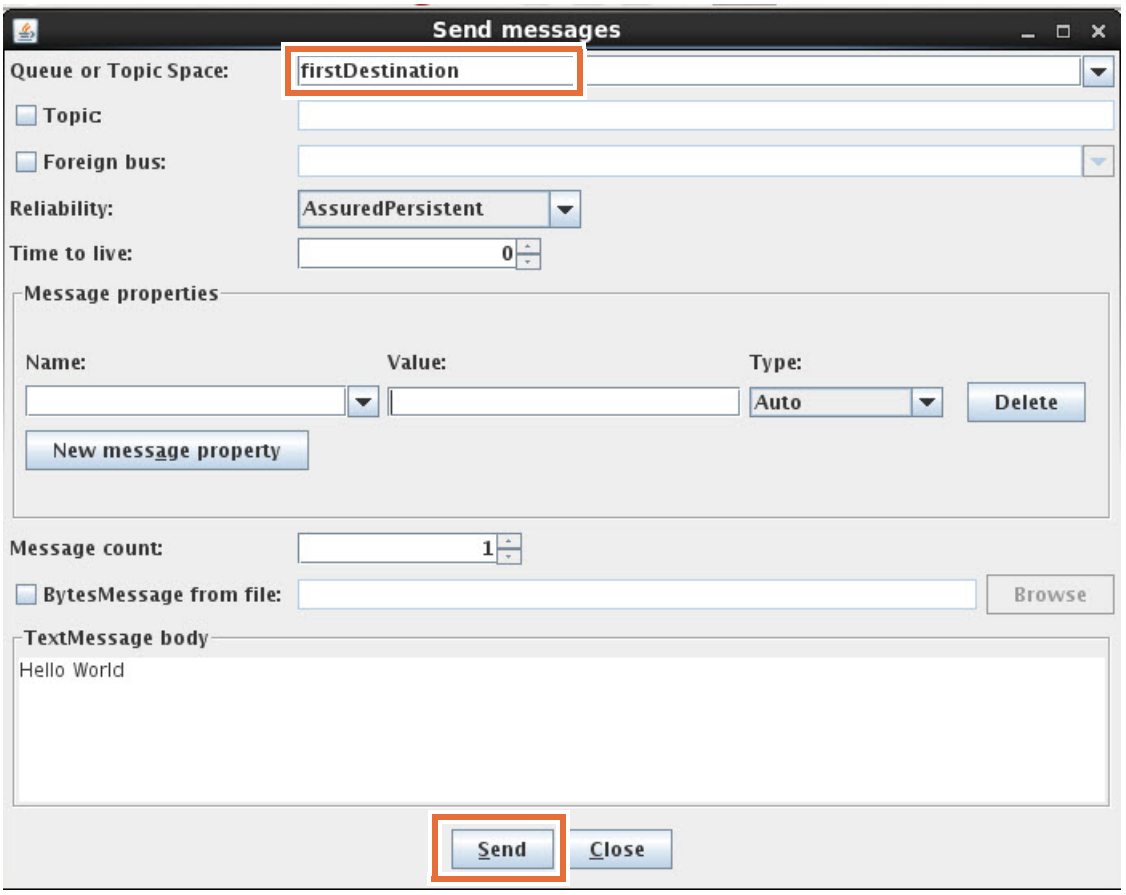
__ e. A **Send messages** dialog box pops up. In the **Queue or Topic space** field type: `firstDestination`

__ f. In the **TextMessage** body, delete what is there and type: `Hello World`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ g. Click **Send**.



__ h. Click **OK** on the **Success box**.

__ i. Click **Close** on the **Send messages** panel.

Section 6: Did the message flow as expected?

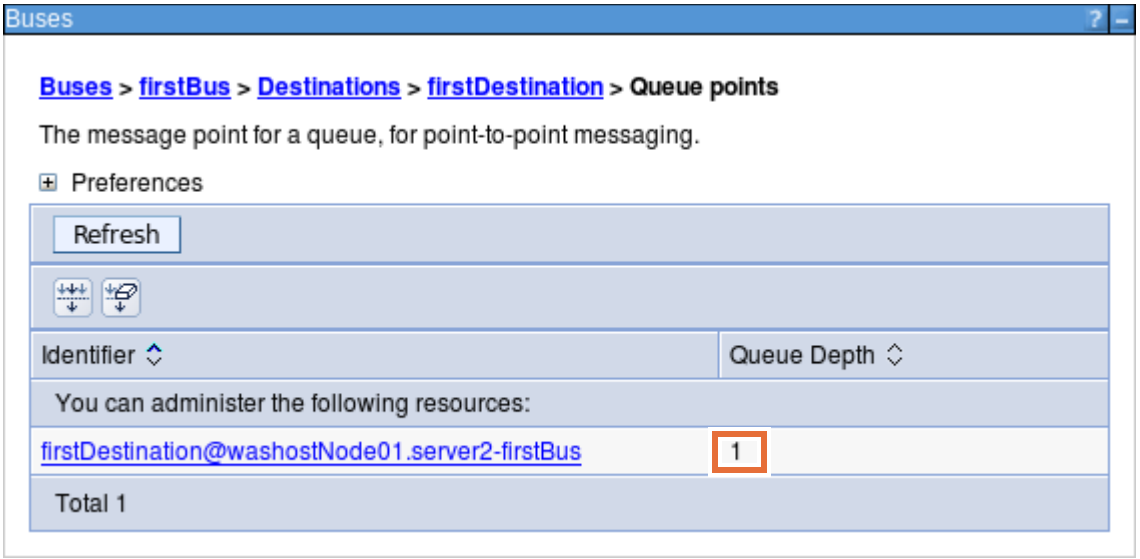
Before you proceed to consume the message, you need to answer the question, “Did the message flow properly?”

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Remember that the destination is not on the messaging engine on which you connected the tool. The destination queue, firstDestination, is hosted on the server2 messaging engine.

__ 1. Verify the location of the message.

__ a. From the administrative console, click **Service integration > Buses > firstBus > Destinations > firstDestination > Queue points**. You might need to click **Refresh**.



__ b. Notice that the message is now on the queue point that is in the messaging engine that runs on **server2**. The Queue Depth value of **1** indicates that there is one message on the queue. This information demonstrates that a client can connect to any bus member, in this case server1, and access any destination that another bus member hosts, in this case server2.

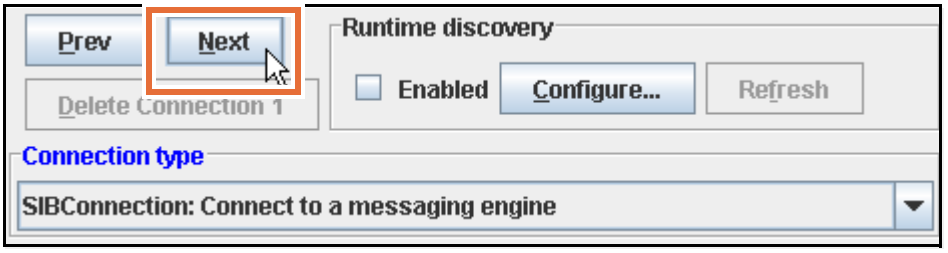
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

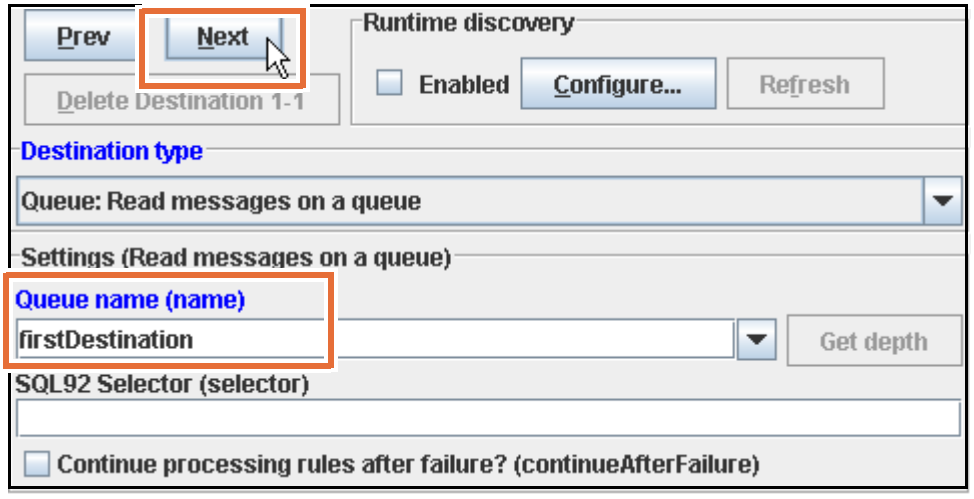
Section 7: Consume the message by using the IBM Service Integration Bus Destination Handler

__ 1. Get the message from the **firstDestination** queue.

__ a. Back in the IBM Service Integration Bus Destination Handler, on the upper right side of the GUI, click **Next**.

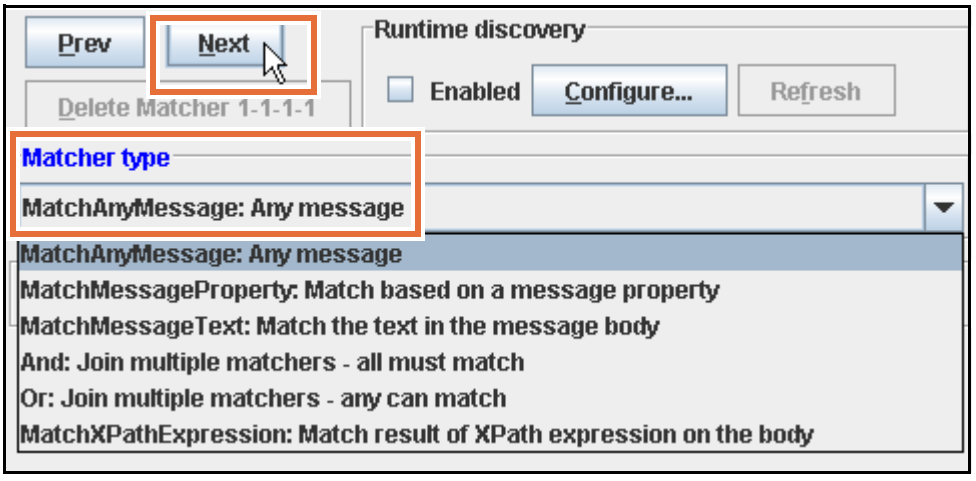


__ b. In the **Queue name** field, type `firstDestination` and click **Next**.



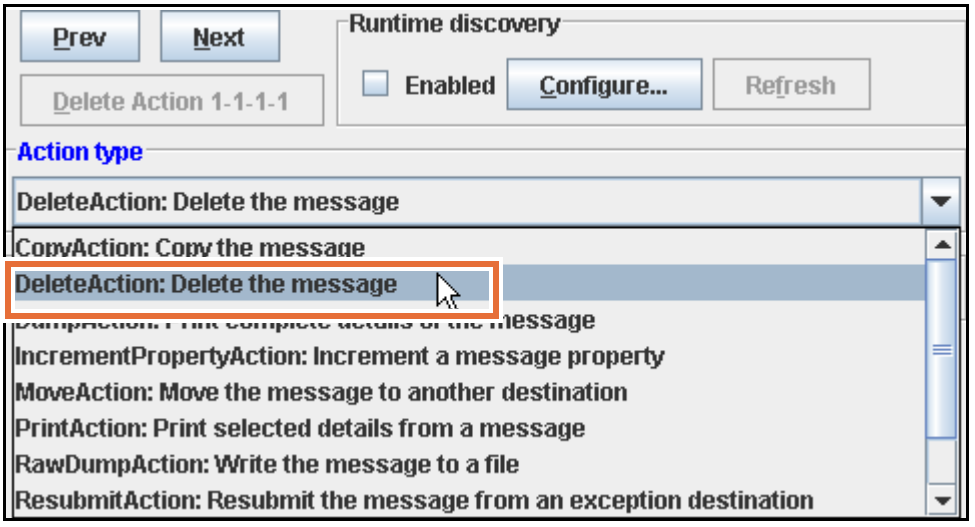
__ c. Click **Next** again.

__ d. In the **Matcher type** menu, select **MatchAnyMessage: Any message** and click **Next**.

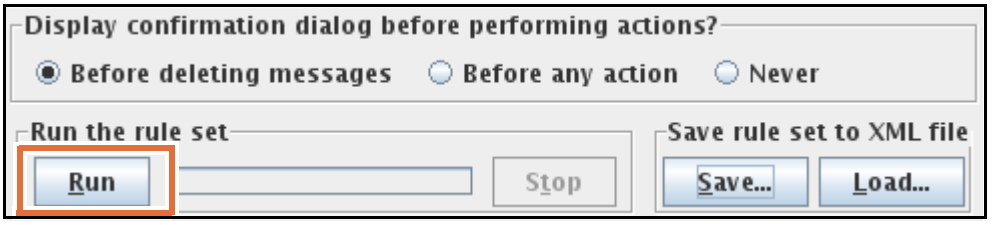


Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

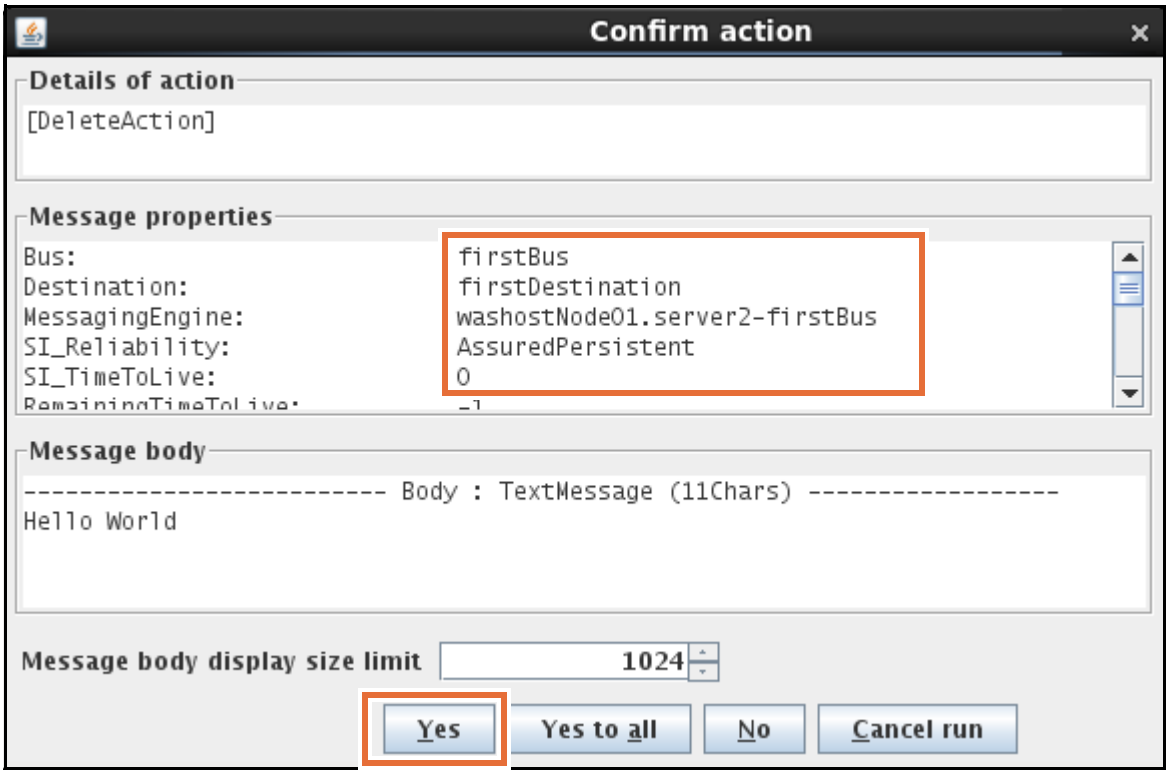
__ e. In the **Action type** menu, select **DeleteAction: Delete the message**.



__ f. In the lower left side of the GUI, click **Run**.



__ g. You see your message properties and message body.



__ h. Notice that the message was consumed from server2.

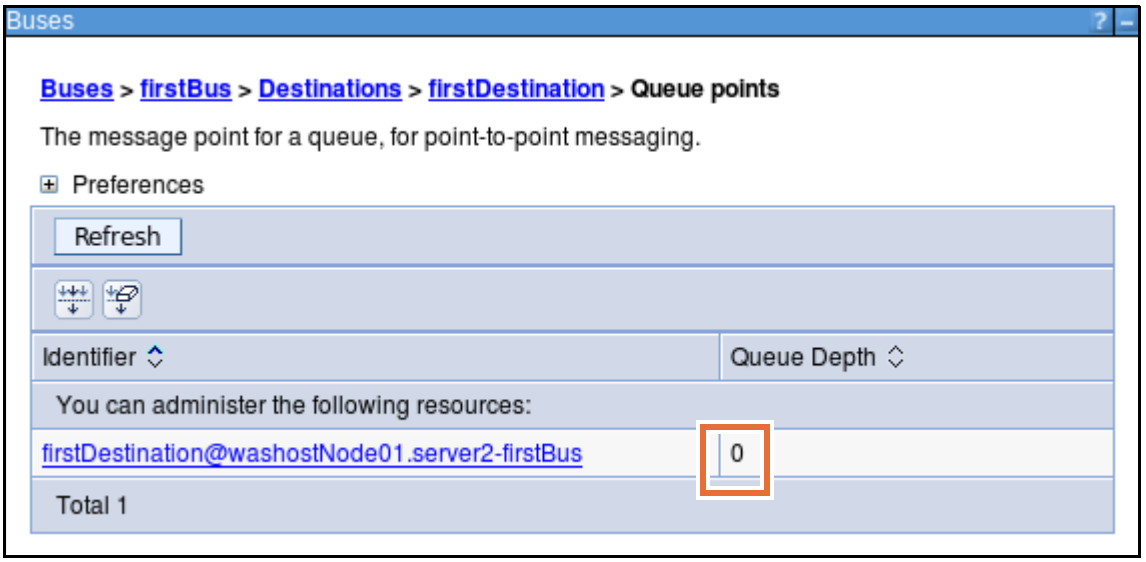
© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ i. In the **Confirm action** panel, click **Yes**.

__ 2. Confirm that the message is removed from the queue point.

__ a. From the administrative console, click **Service integration > Buses > firstBus > Destinations > firstDestination > queue points**. If the console is already open to this view, click **Refresh**.

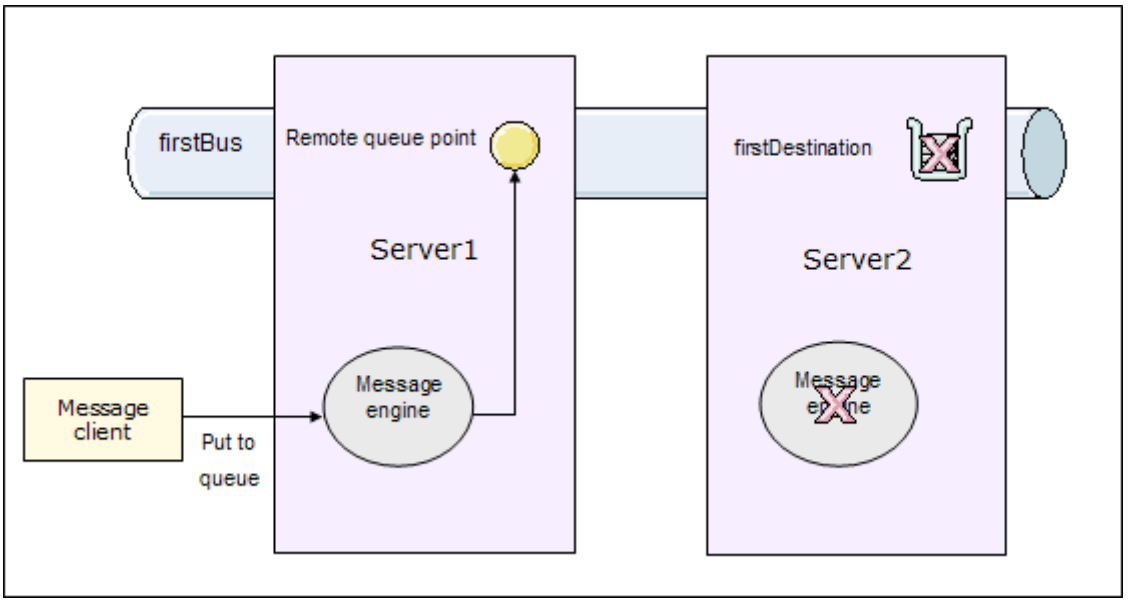


__ b. Notice that the message is gone from the queue point as the **Queue Depth** is now zero.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 8: Explore message flow when an ME hosting a destination is down

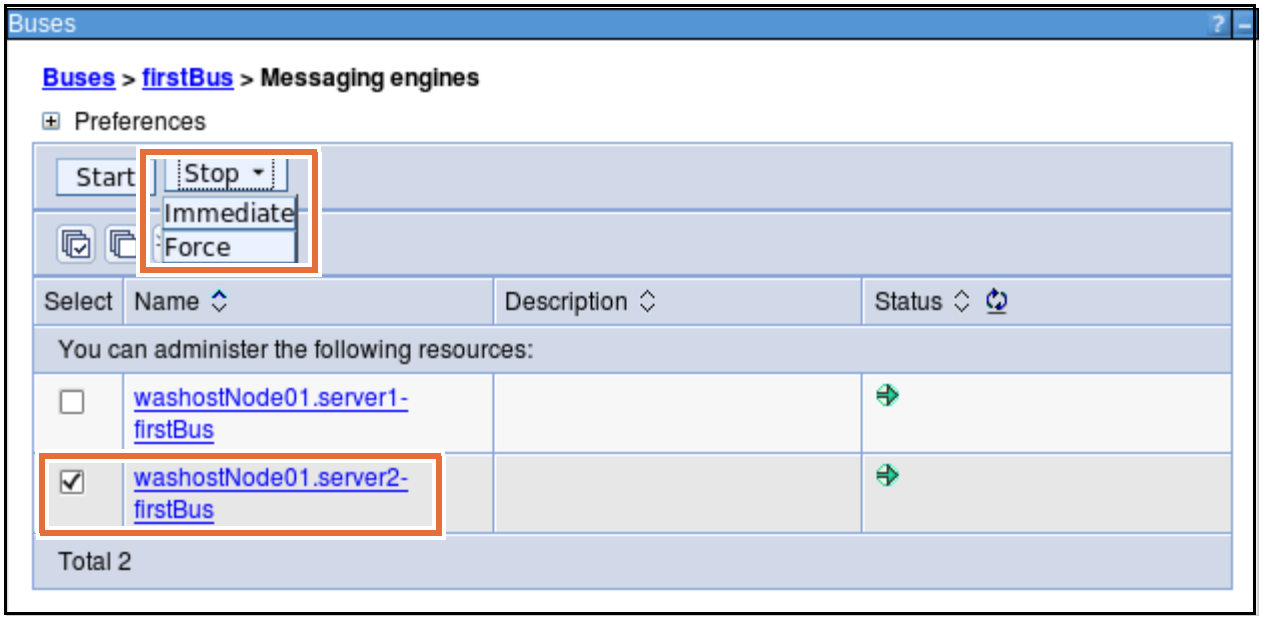
In this part of the exercise, you stop the messaging engine on server2 and then send a message to the queue firstDestination. You explore the message flow by using the administrative console.



__ 1. Shut down the messaging engine that hosts the queue point for firstDestination.

__ a. From the administrative console, click **Service integration > Buses > firstBus > messaging engines**.

__ b. Click the check box next to the messaging engine for **server2** and click **Stop > Immediate**.



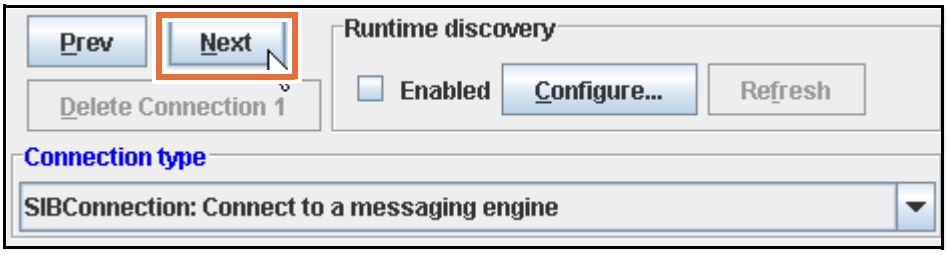
__ c. Wait for the messaging engine to stop.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 2. Produce another message.

__ a. Go back to the **IBM Service Integration Bus Destination Handler**, in the upper right GUI, and click **Next**.



__ b. Click **Send message** again, but this time type in the TextMessage body: `Will I make it to firstDestination`?

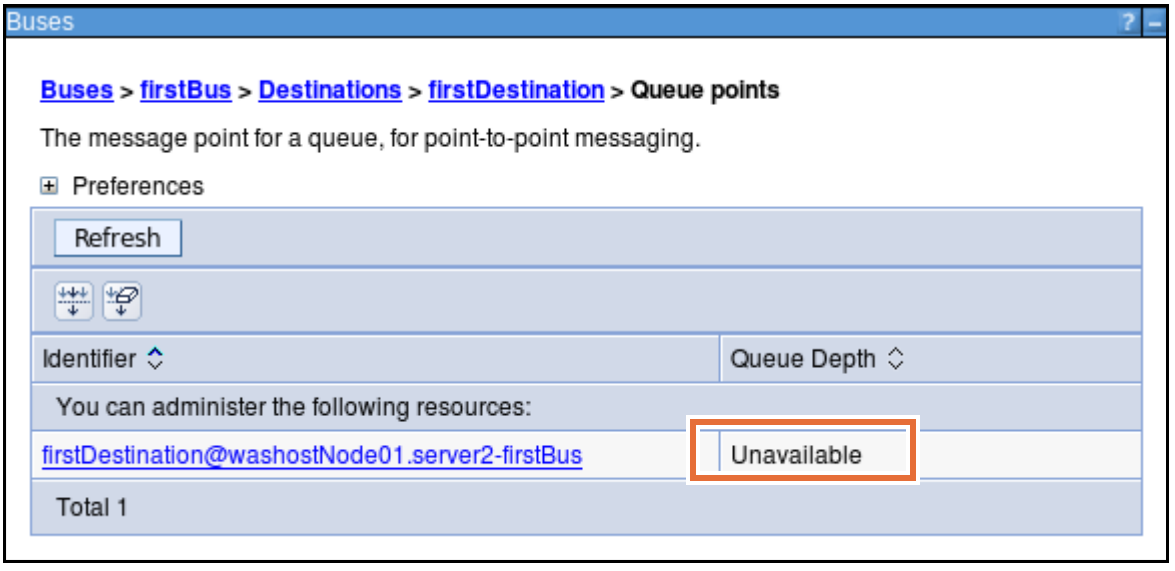
__ c. Click **Send**.

__ d. Click **OK**.

__ e. Click **Close** on the **Send messages** panel.

__ 3. Verify whether the message flowed as expected.

__ a. From the administrative console, click **Service integration > Buses > firstBus > Destinations > firstDestination > Queue points**.



__ b. Notice that the state of the queue is **unavailable**. The queue is unavailable because the hosting messaging engine server2 is not running.

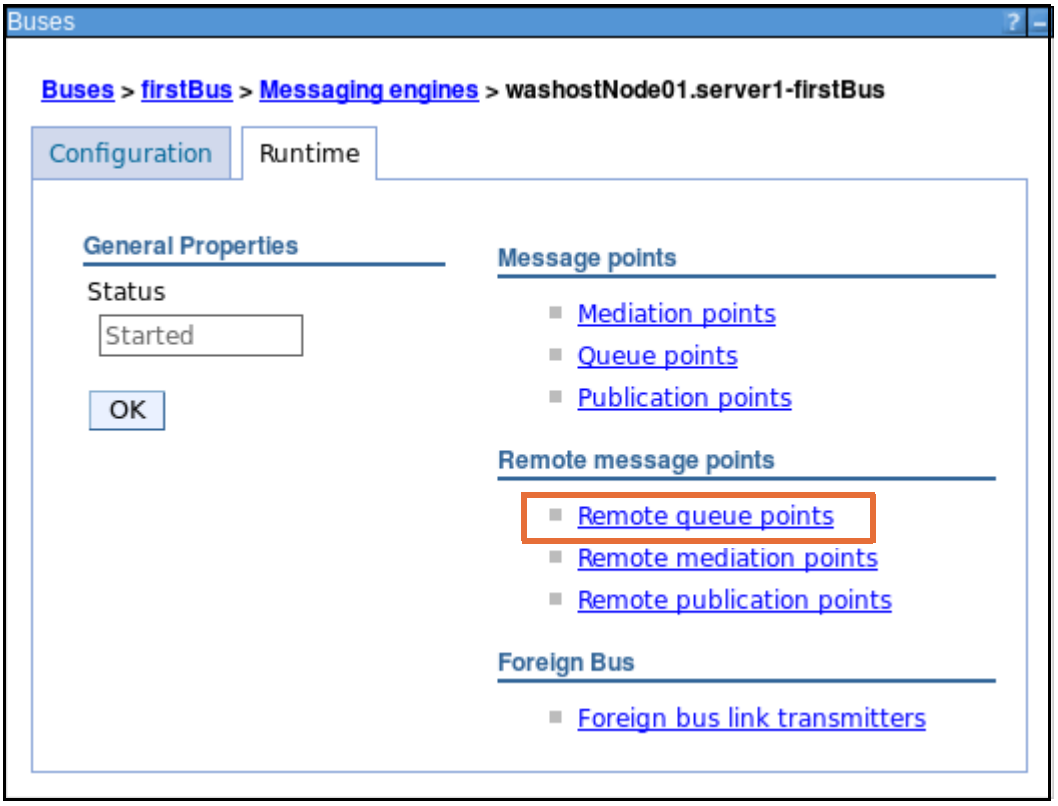
__ 4. Where is the message? The production of the message succeeded. You put the message to the bus, but it is not on the destination to which you wanted to send it. Try to locate the message.

__ a. Click **Service integration > Buses > firstBus > Messaging engines**.

__ b. Click the link for the one running messaging engine, **server1-firstBus**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select the **Runtime** tab.



__ d. Click **Remote queue points**.

__ e. You can see that the message is on the bus (one current outbound message for firstDestination). It is waiting for the messaging engine that hosts the queue point for firstDestination to become available. The message is on a remote queue point, and it is waiting to be delivered to the messaging engine on server2.

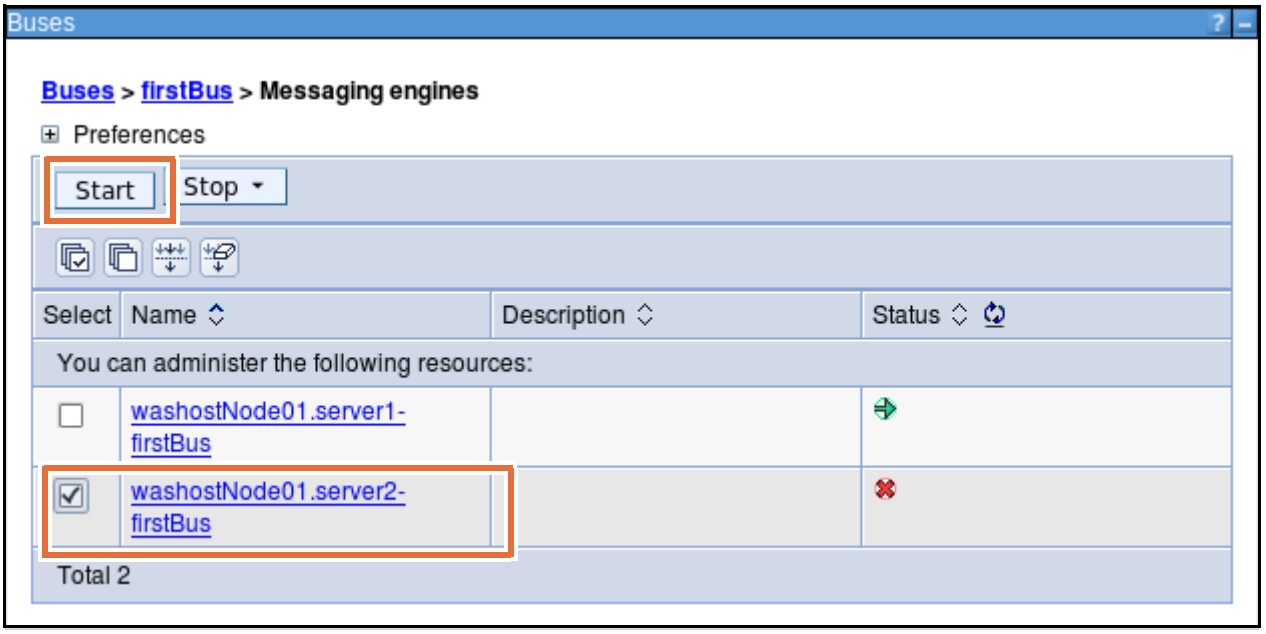
 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ 5. Restart the messaging engine on server2 and check whether the message flows properly.

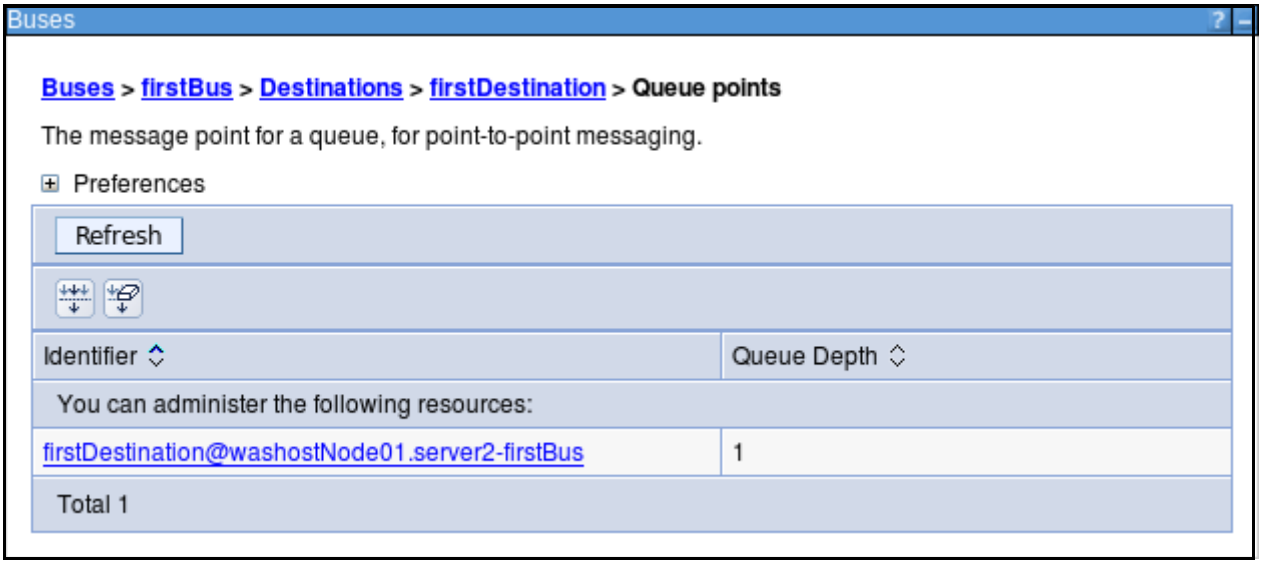
__ a. Click **Service integration > Buses > firstBus > Messaging engines**.

__ b. Click the check box next to the messaging engine for **server2** and click **Start**.



__ c. Wait for the messaging engine to start.

__ d. Click **Service integration > Buses > firstBus > Destinations > firstDestination > Queue points**.



__ e. Notice that the message is now on the firstDestination queue point where it was intended to be.

__ 6. As you did before, check the Remote queue points for the messaging engine on server1.

__ a. Click **Service integration > Buses > firstBus > Messaging engines**.

__ b. Click the link for the messaging engine on server1, **server1-firstBus**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Select the **Runtime** tab.

__ d. Click the link for **Remote queue points**.

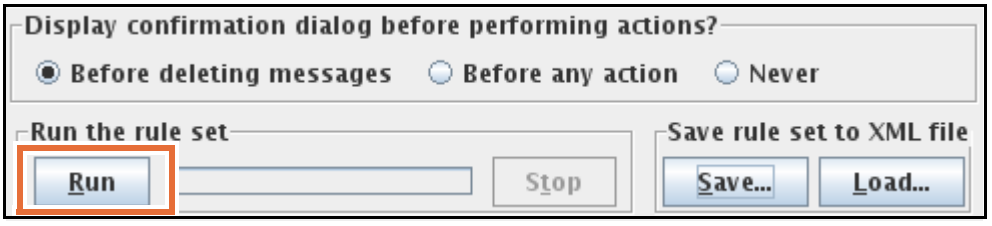
__ e. Notice that there are no longer any outbound messages for firstDestination on the queue point of server1.

__ f. Log out of the administrative console.

__ 7. Consume the message from **firstDestination**.

__ a. Go back to the IBM Service Integration Bus Destination Handler panel.

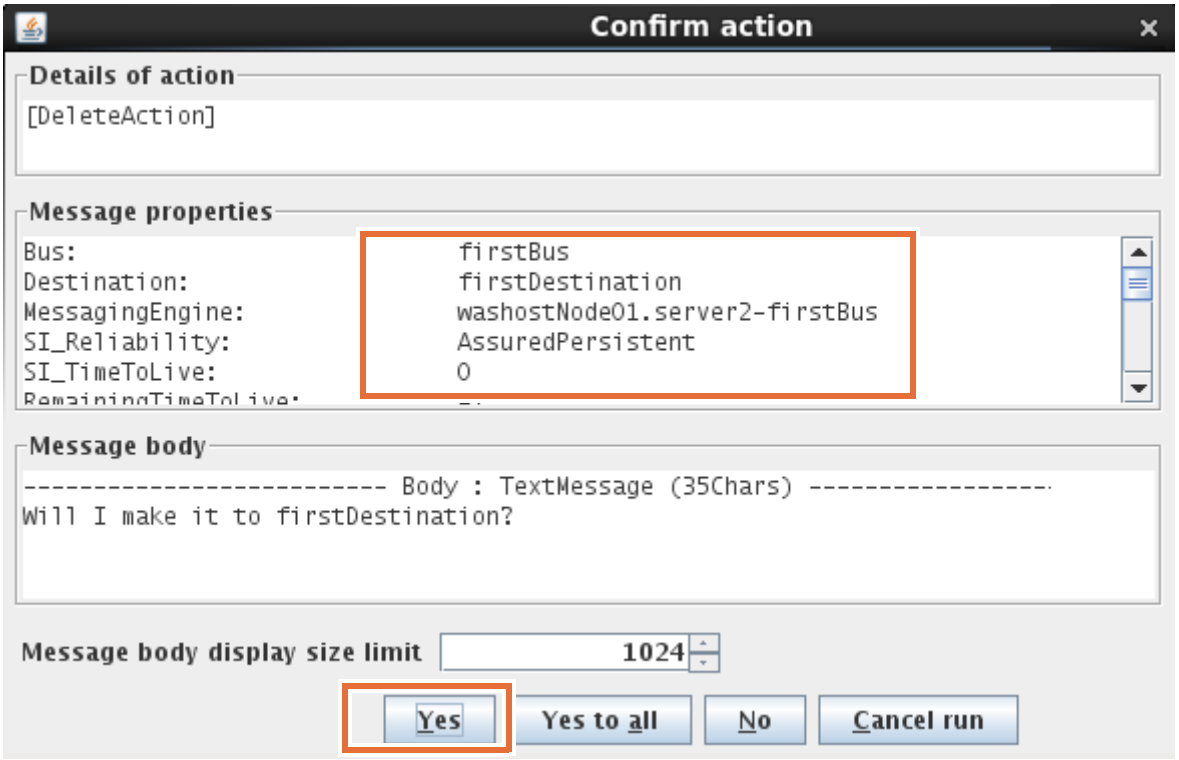
__ b. In the lower left GUI, click **Run** to consume the message.



© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. You see the message properties and message body.



__ d. In the **Confirm action** panel, click **Yes**.

__ e. Close the IBM Service Integration Bus Destination Handler.

END OF LAB

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

In this lab exercise, you learned how to define a simple two-member bus. You also experienced some common issues when trying to diagnose message flow issues. You worked through the issues by looking in the administration panels and by using the IBM Service Integration Bus Destination Handler.

To find the tool that is used in this lab, go to the following website:

- IBM Service Integration Bus Destination Handler:

  `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24021439`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise 13.Configuring health management policies

What this exercise is about

This exercise shows you how to configure the environment to monitor the health of application servers. Requests are sent to an application. The requests trigger the violation of a health policy. You also see how WebSphere Application Server Intelligent Management implements a defined action plan when the health policy is violated.

What you should be able to do

At the end of this exercise you should be able to:

- Configure a health policy to monitor application servers for the violation of a health condition
- Configure the type of violation to look for
- Configure actions to take when a violation occurs
- Test the health policy by sending requests to the application that trigger the violation of the health policy
- Examine the task that gets generated when a health policy is violated
- Examine the results when the task is completed

Introduction

WebSphere Application Server Intelligent Management provides many advanced capabilities that can be used to monitor and manage high performance production environments. These capabilities include the ability to monitor the environment for specific conditions that indicate unhealthy servers. Intelligent Management can be configured to take specific actions when a health condition is violated. The actions can be taken automatically, or after review and approval by an administrator.

In this exercise, you create a policy that monitors server1. You build an action plan that takes effect when the policy is violated. You review the task that is generated when the policy is violated. You review the results of the health policy action plan actions.

Requirements

This exercise requires that the BadAppEARProject application is installed and its modules are mapped to server1.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise instructions

Note

`<profiles_root> = /opt/IBM/WebSphere/AppServer/profiles`

Section 1: Install the BadAppEARProject application if required

For this lab exercise, the BadAppEARProject application is required. Uninstallation of the BadAppEARProject application might be part of the cleanup section of a previous lab. If that is the case, you must reinstall it for this lab exercise.

__ 1. The BadApp application is installed in Exercise 4. If you did not complete that exercise, install the BadAppEARProject application.

__ a. Open a terminal window and change to directory: `<profiles_root>/Dmgr/bin`

__ b. Run the command:

```
./wsadmin.sh -f /usr/Student/Hang/install_badapp.py -user wasadmin
-password web1sphere
```

__ c. Make sure that the command completes successfully and you see the message:

```
Application BadAppEARProject installed successfully.
```

Section 2: Configure settings for exercise environment

Important

Before beginning this section, verify that you completed the step to edit the `/etc/hosts` file and add the IP address of your lab image. This step is done in **Exercise 2. Gathering diagnostic data**. **Section 1**, **Step 1**.

Because you are working in a lab exercise environment, some configuration settings must change. To complete the lab in a reasonable amount of time, you specify a time setting value that would typically not be used in a real world environment. You also specify a configuration setting for the BadAppEARProject application so that the application does not fail too quickly.

__ 1. Log in to the administrative console of the deployment manager.

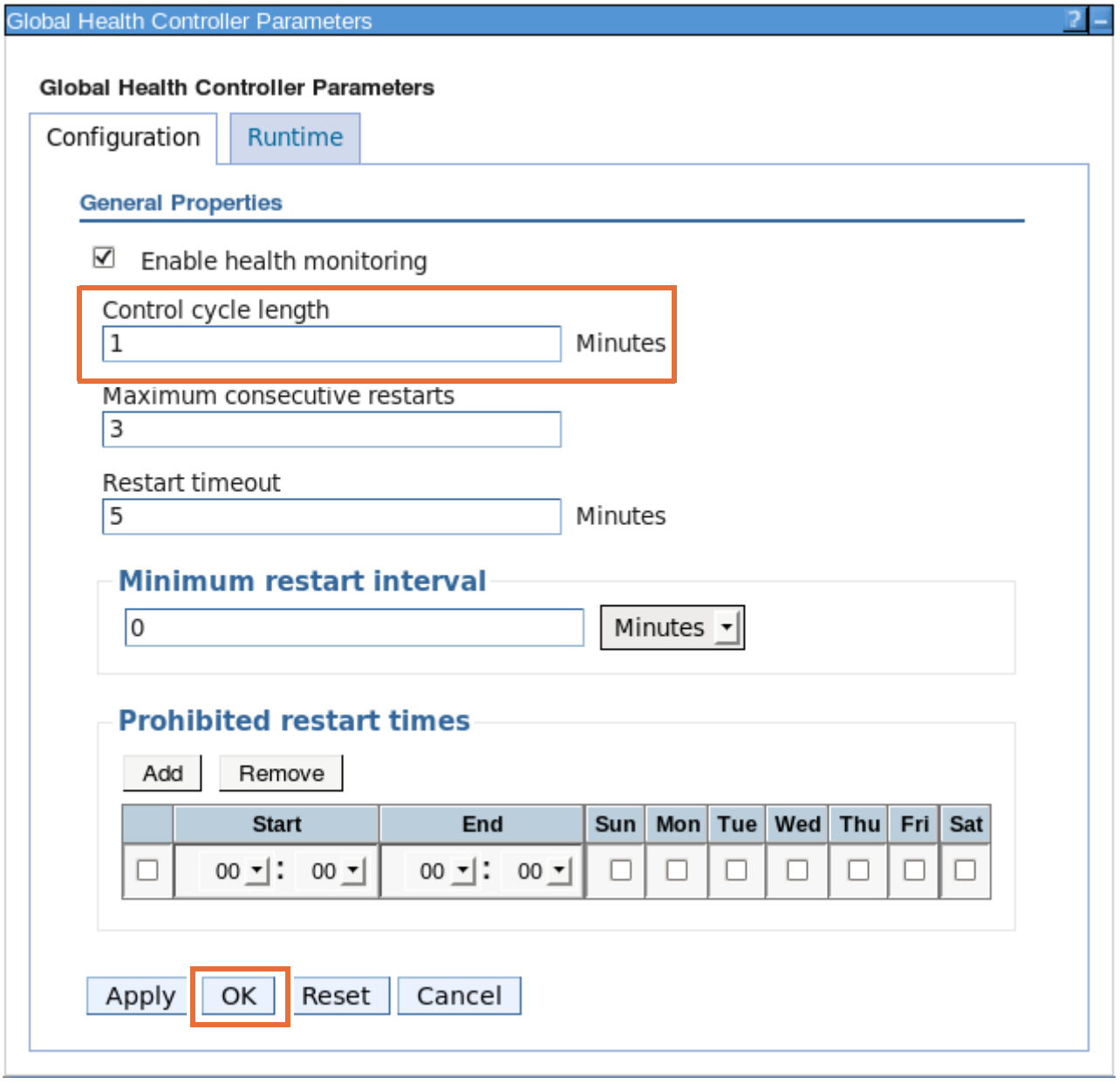
__ 2. Set the health controller to react more quickly.

By default, the health controller checks all health conditions every 5 minutes. You change the check interval to 1 minute so that you do not have to wait so long for the health controller to react to a violation of a health policy.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ a. In the administration console, select **Operational policies > Autonomic Managers > Health Controller**.

__ b. Change the health controller **Control cycle length** to **1** Minutes.



__ c. Click **OK**, and **Save** directly to the master configuration.

__ 3. Configure the BadAppEARProject application so that it does not fail too quickly.

The default settings for the BadAppEARproject use up memory too quickly for the health controller to react before the application server fails. If an application causes a server to fail within a minute or two in the real world, it would most likely be detected in a test environment, rather than in production. So having the application take longer to fail is more realistic.

__ a. In the administration console, select **Servers > Server types > WebSphere application servers**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Click **server1**.

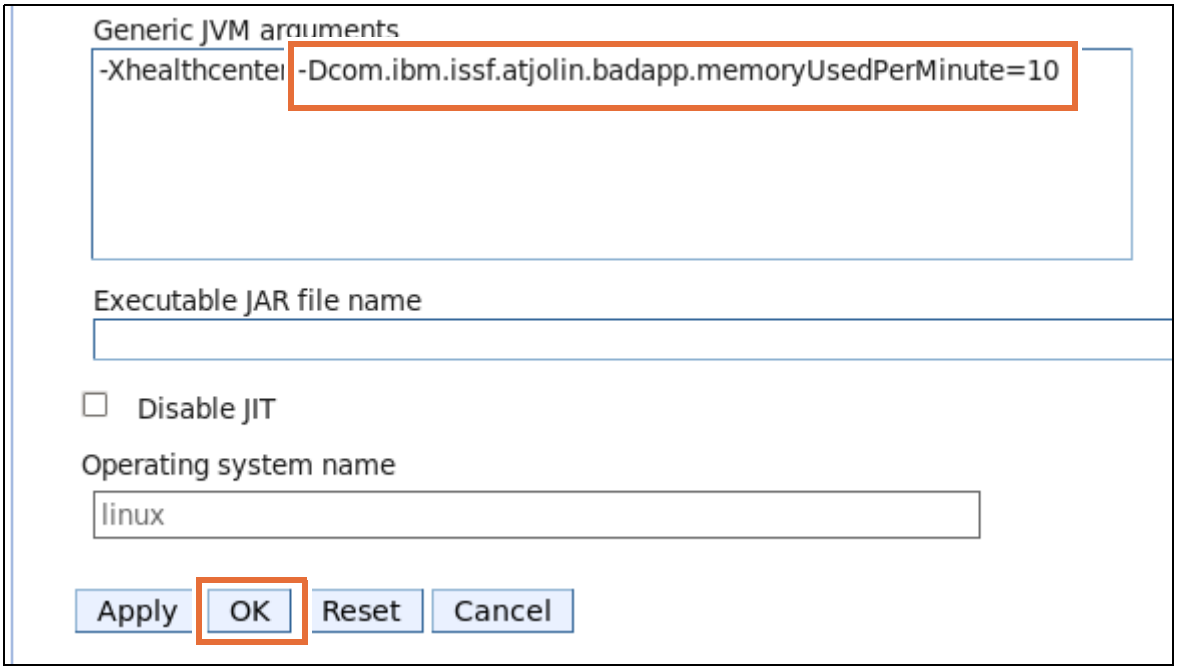
__ c. On the right side, under **Server Infrastructure**, expand **Java and Process Management**.

__ d. Select **Process definition**.

__ e. On the right side, under **Additional Properties**, select **Java Virtual Machine**.

__ f. In the **Generic JVM arguments** field, add the argument:

```
-Dcom.ibm.issf.atjolin.badapp.memoryUsedPerMinute=10
```



__ g. Click **OK**, and **Save** directly to the master configuration.

Note

Memory that is used per minute argument

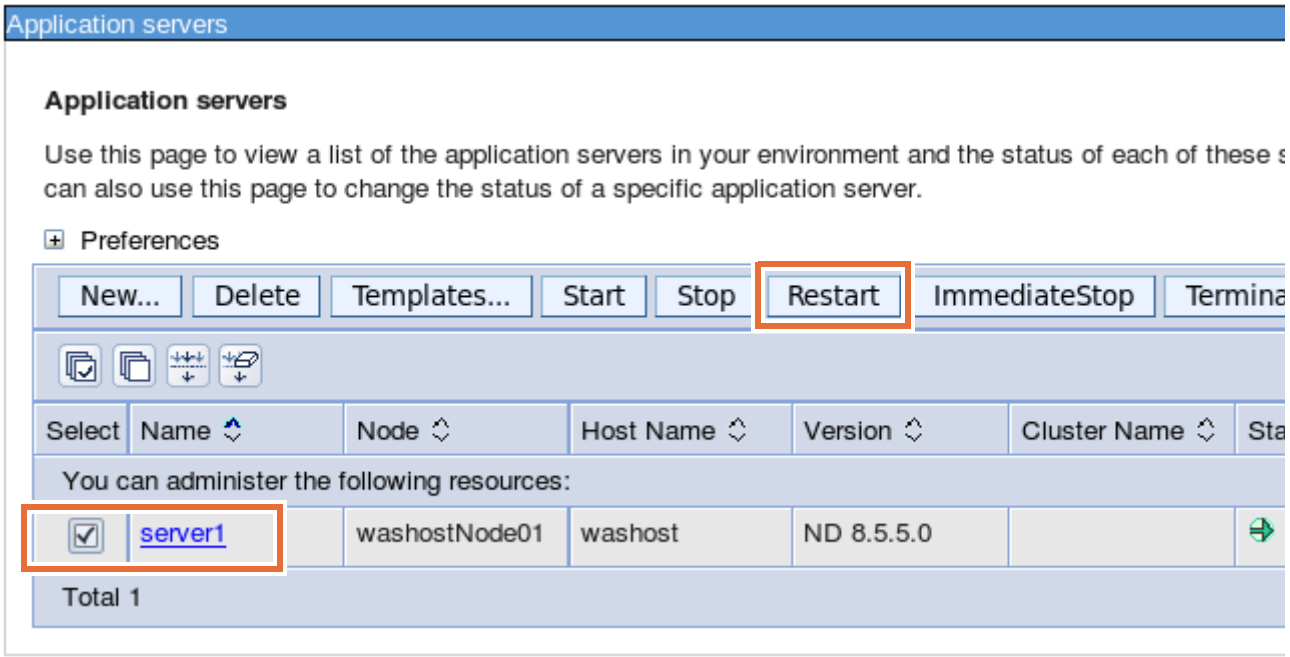
This argument specifies how much memory (in Megabytes) is used up in 1 minute by the BadApp application, under behavior types 4 or 5. With the default value of 35, the server becomes unresponsive in just a few minutes. The value is set to 10 so that the server can still respond to heap dump and restart actions.

__ 4. Restart server1.

__ a. From the breadcrumb trail, select **Application servers**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Select **server1**, and click **Restart**.



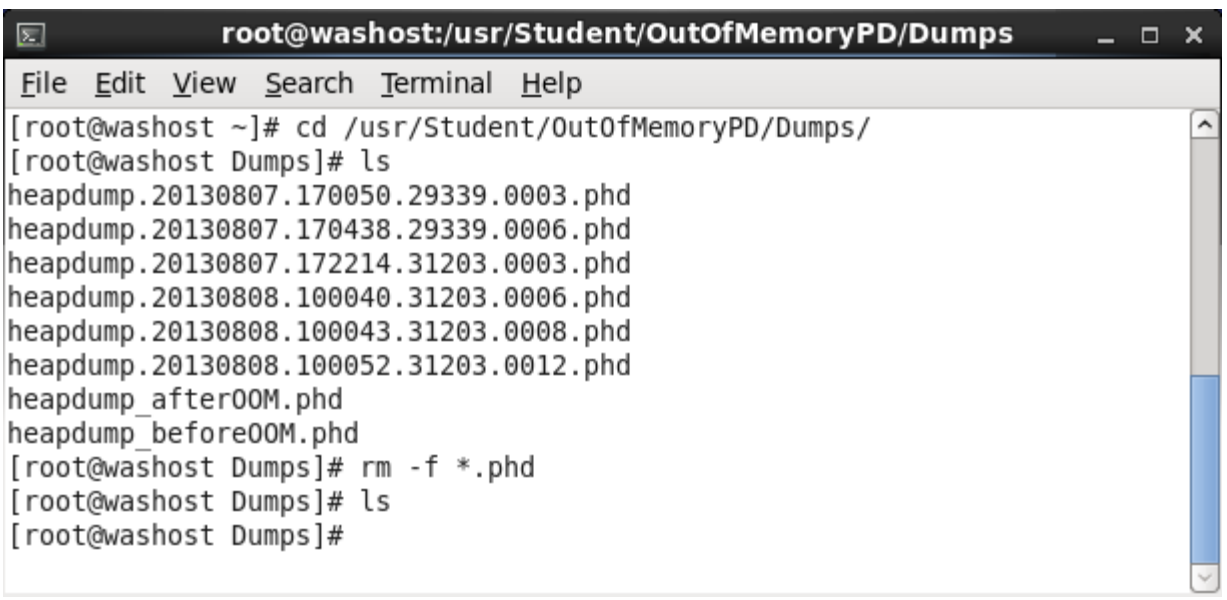
__ c. Wait for the server to restart.

__ 5. Make room for heap dumps generated by health controller actions. By default, the health controller does not generate new heap dumps if 10 heap dumps exist in the heap dump directory.

__ a. In a terminal window, change directories to: `/usr/Student/OutOfMemoryPD/Dumps`

__ b. Remove any existing portable heap dump files with the command:

```
rm -f *.phd
```



 © Copyright IBM Corp. 2013

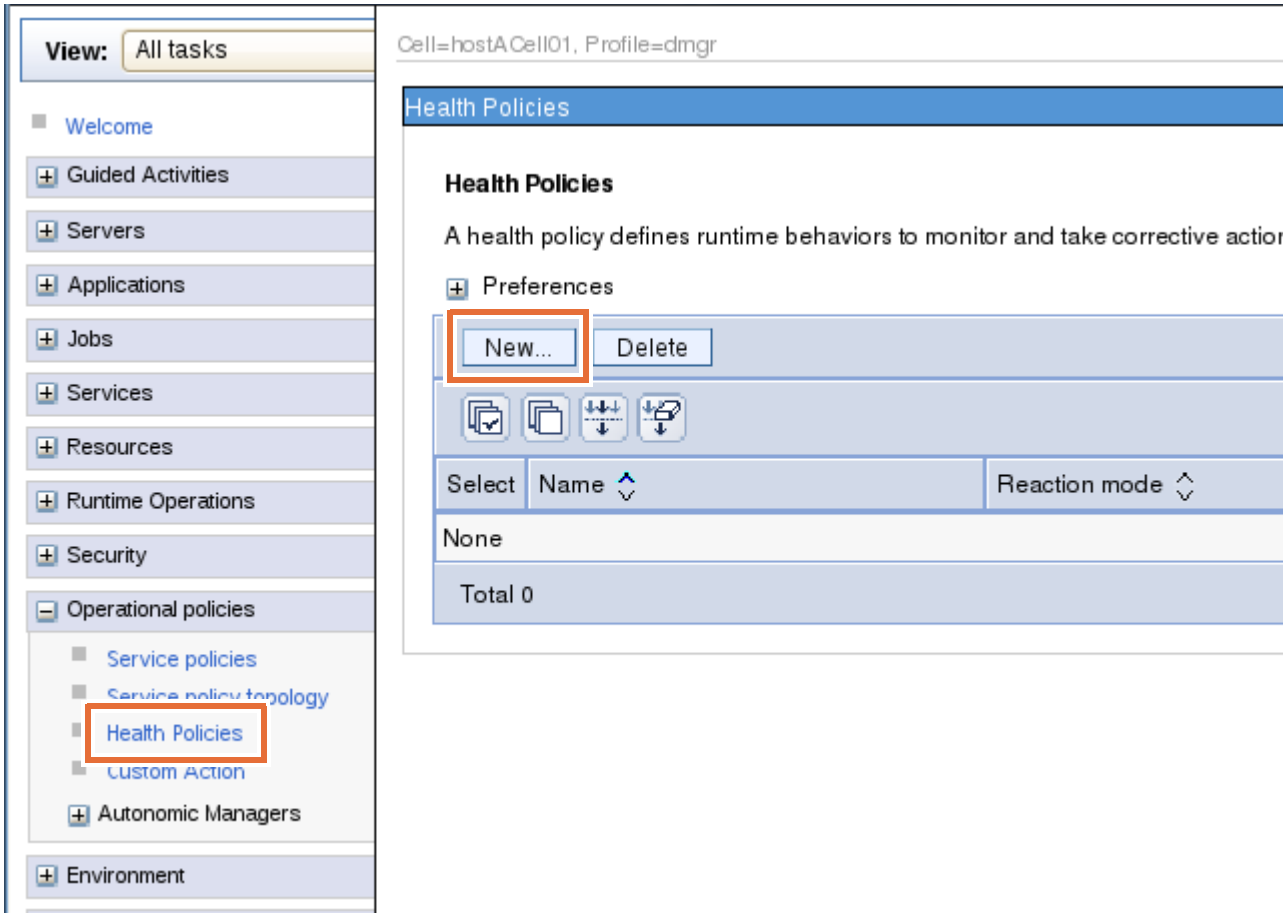
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Section 3: Configure a health policy

Configure a WebSphere Intelligent Management health policy to monitor the health of a server and to perform an action plan when the policy is violated.

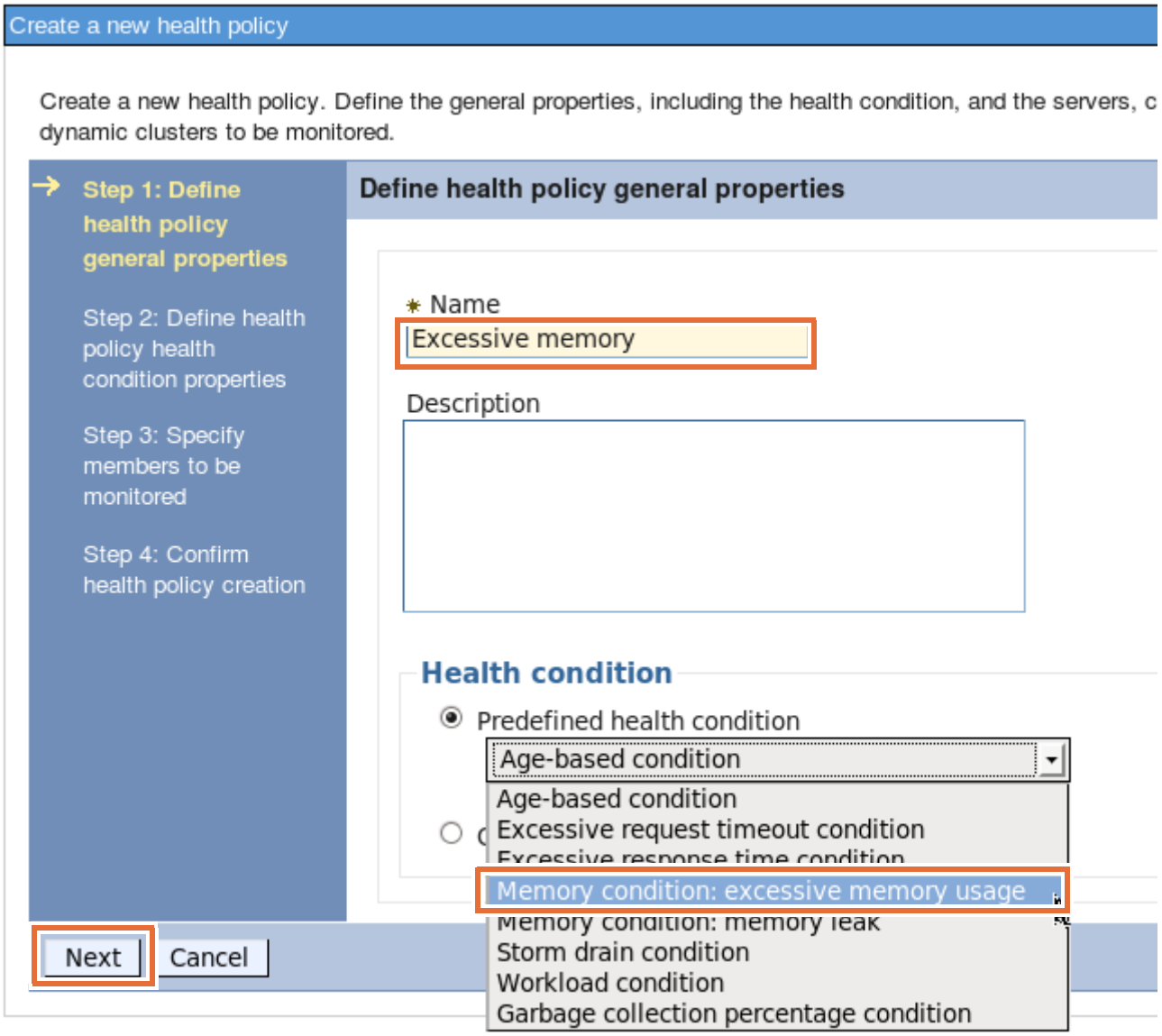
__ 1. Configure a health policy that restarts the server and creates a heap dump when server1 uses too much memory.

__ a. In the administration console, navigate to **Operational policies > Health Policies** and click **New**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. For **Step 1**, give the health policy the name **Excessive Memory**. Notice that many health conditions can be monitored including memory usage, server age, and response times. Choose **Memory condition: excessive memory usage** for Predefined health condition. Click **Next**.

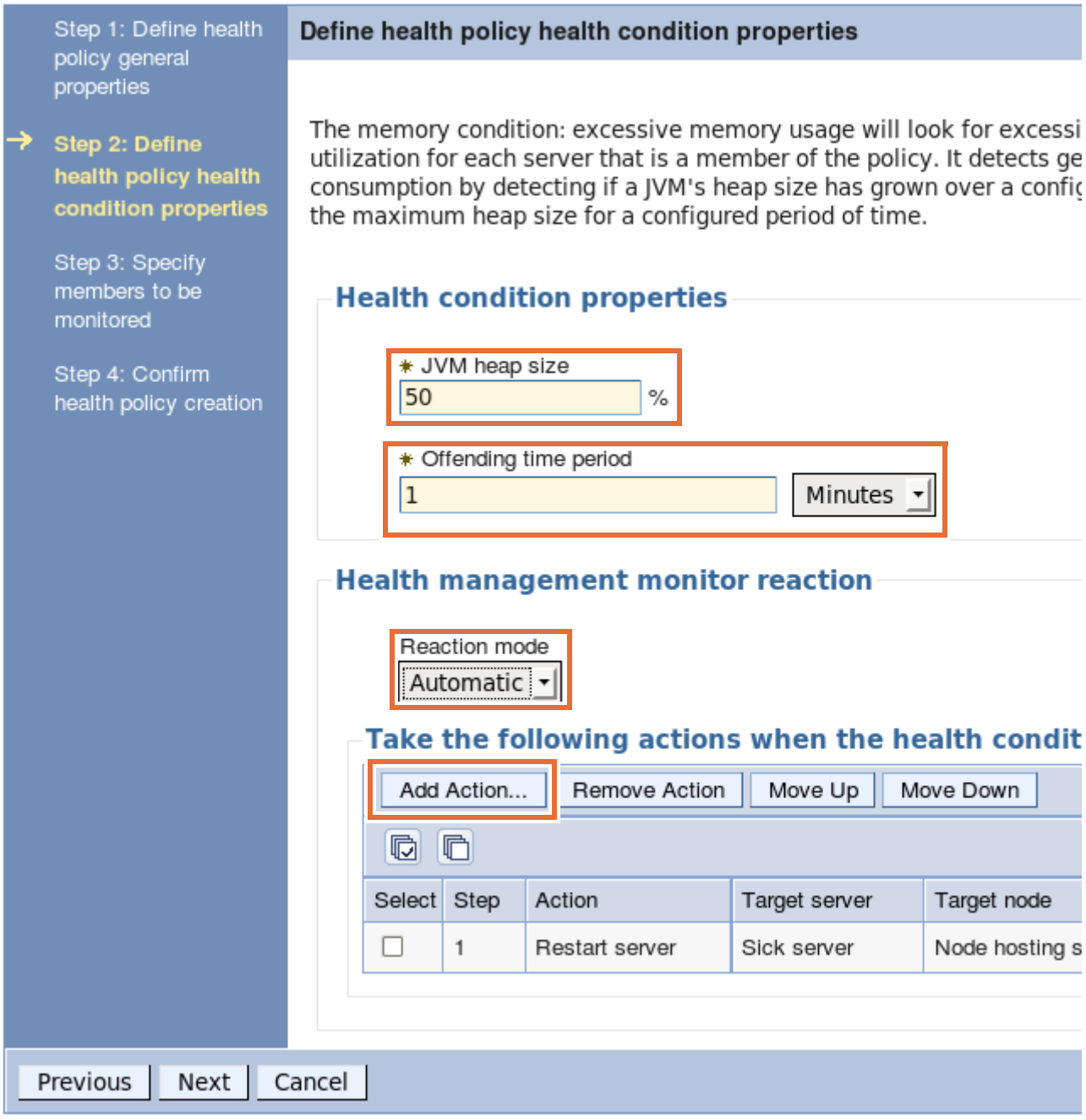


__ c. For **Step 2**, define the health condition properties that set the limits for the monitoring. Set the **JVM heap size** to **50**%. Set the **Offending time period** to **1** Minutes. Change the **Reaction mode** to **Automatic**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Click **Add Action** to define a new action for the health policy.

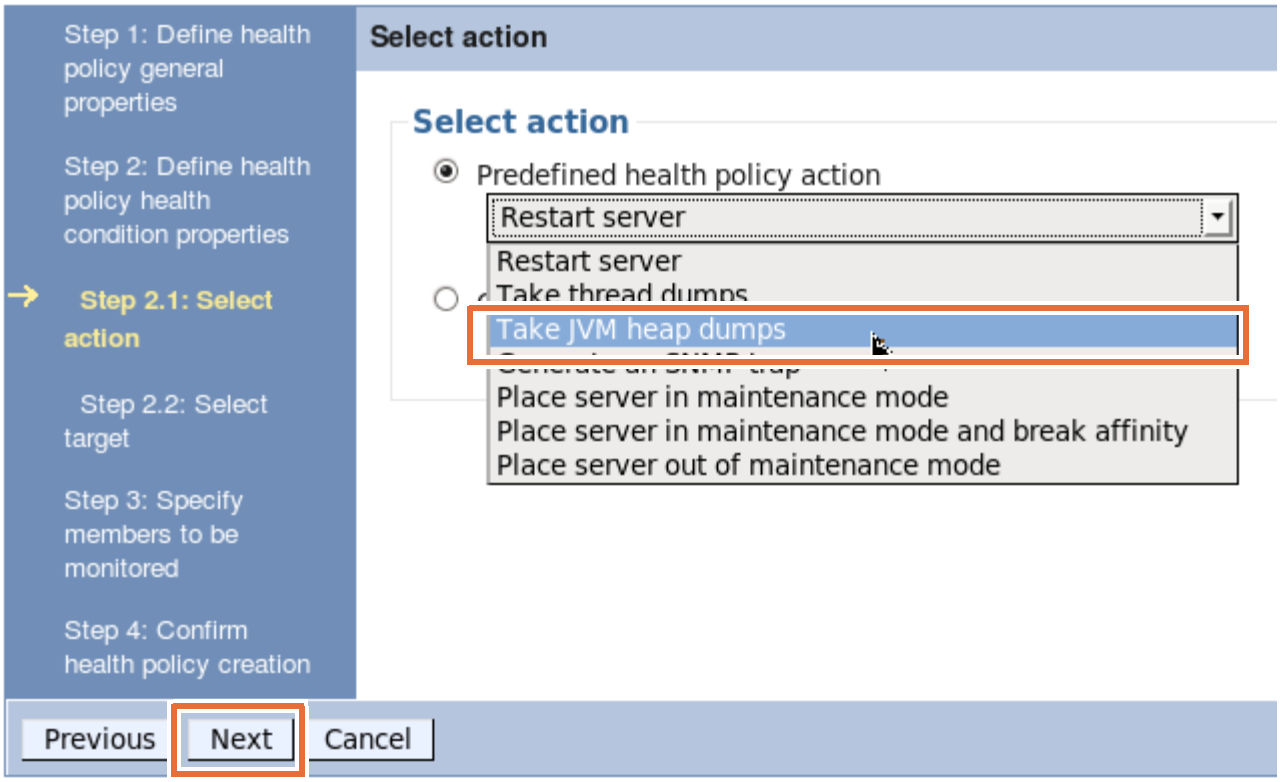


Information

In the real world, you might want to specify a higher JVM heap size percentage for a longer period. Because the application in this exercise uses memory so quickly, these settings allow the health controller to react before the application causes an out-of-memory exception.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

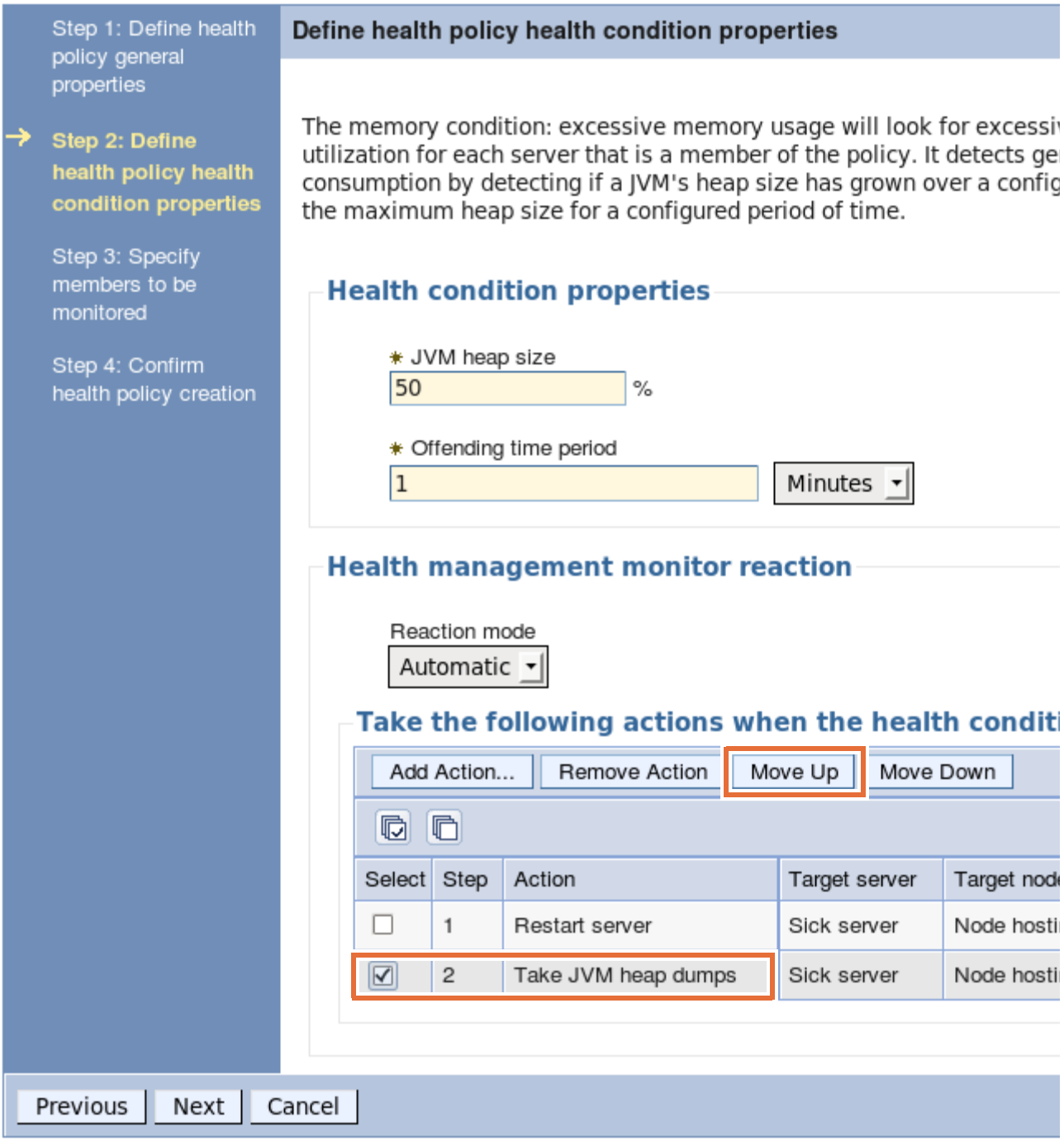
__ e. Select **Predefined health policy action** and then the action value of **Take JVM heap dumps**. Click **Next**.



 © Copyright IBM Corp. 2013

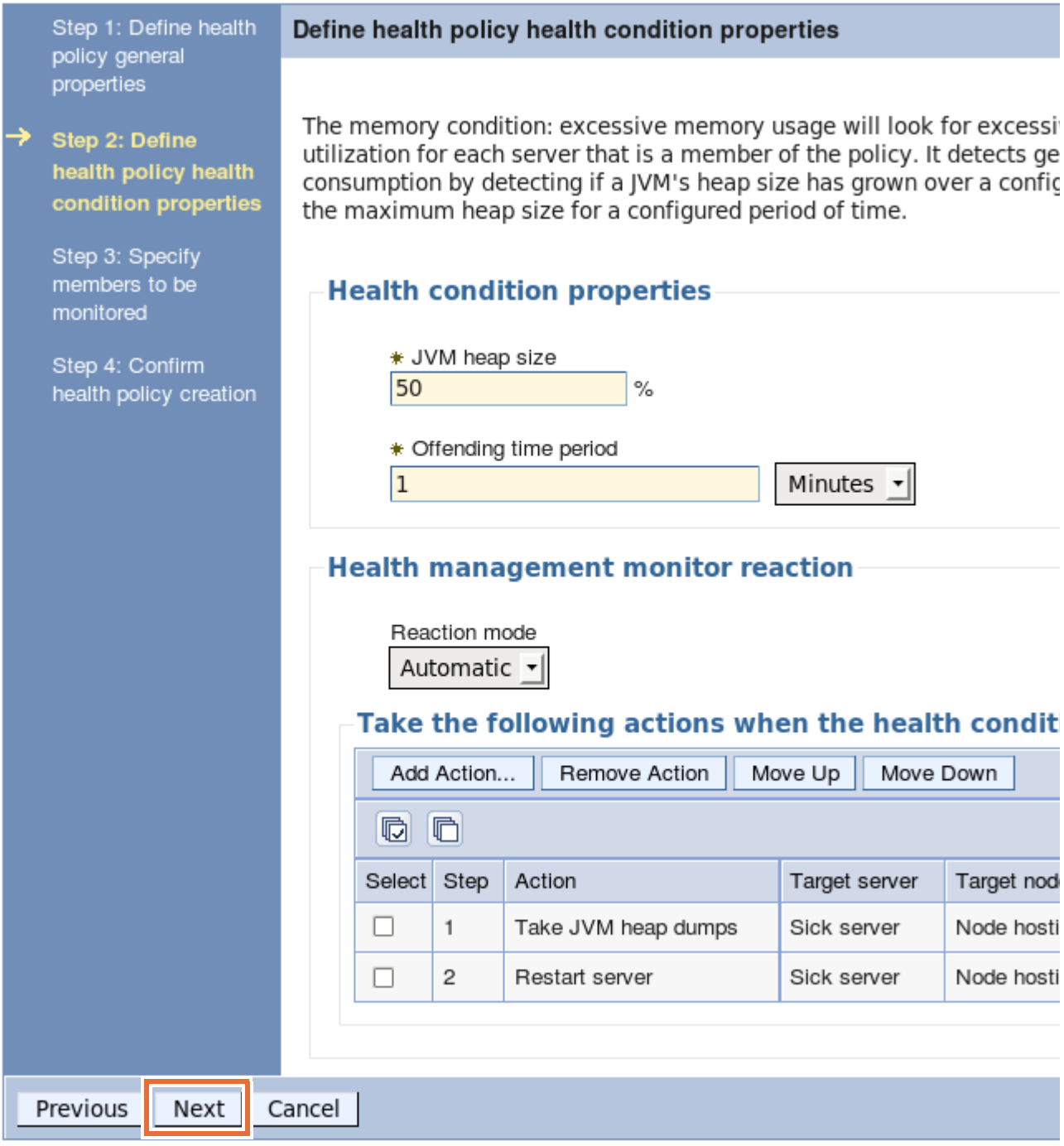
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ f. In the health policy properties panel select action number **2**, **Take JVM heap dumps**, click **Move up**, and click **Next.**



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

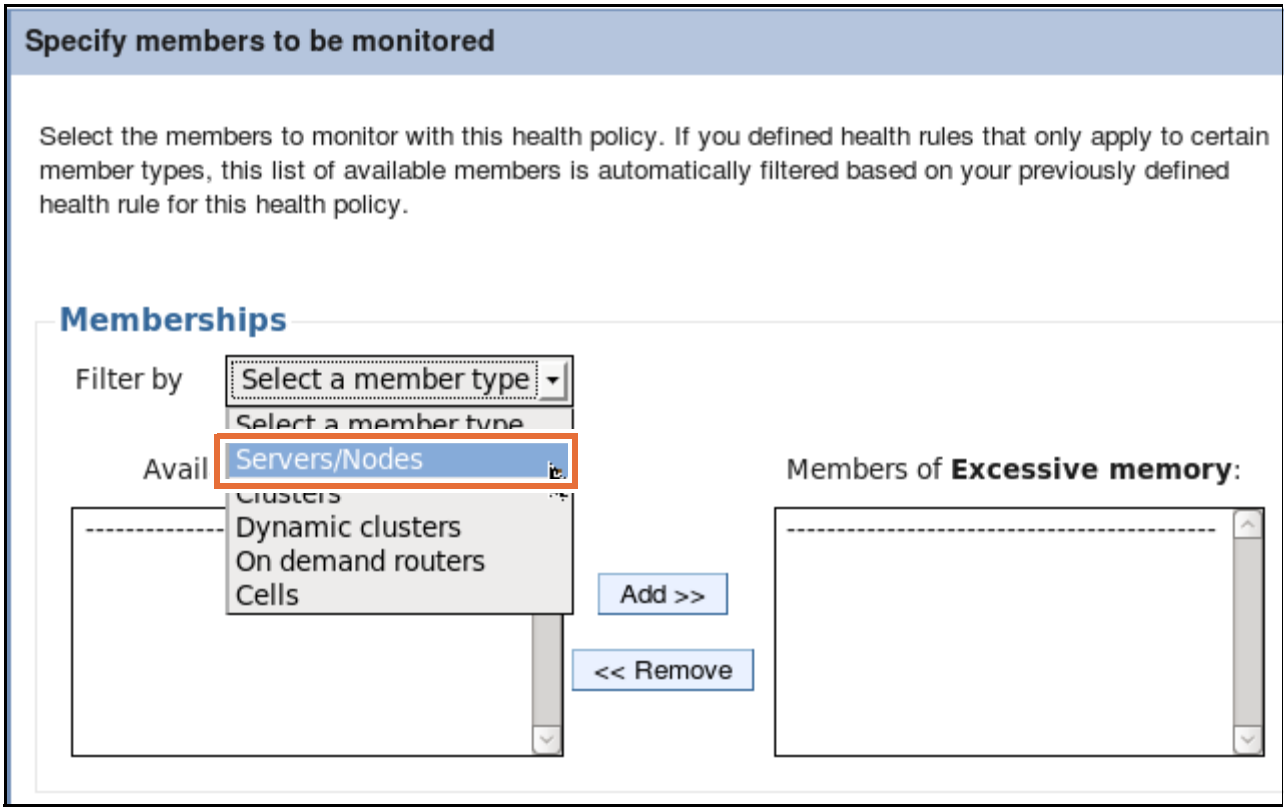
__ g. Click **Next**.



© Copyright IBM Corp. 2013

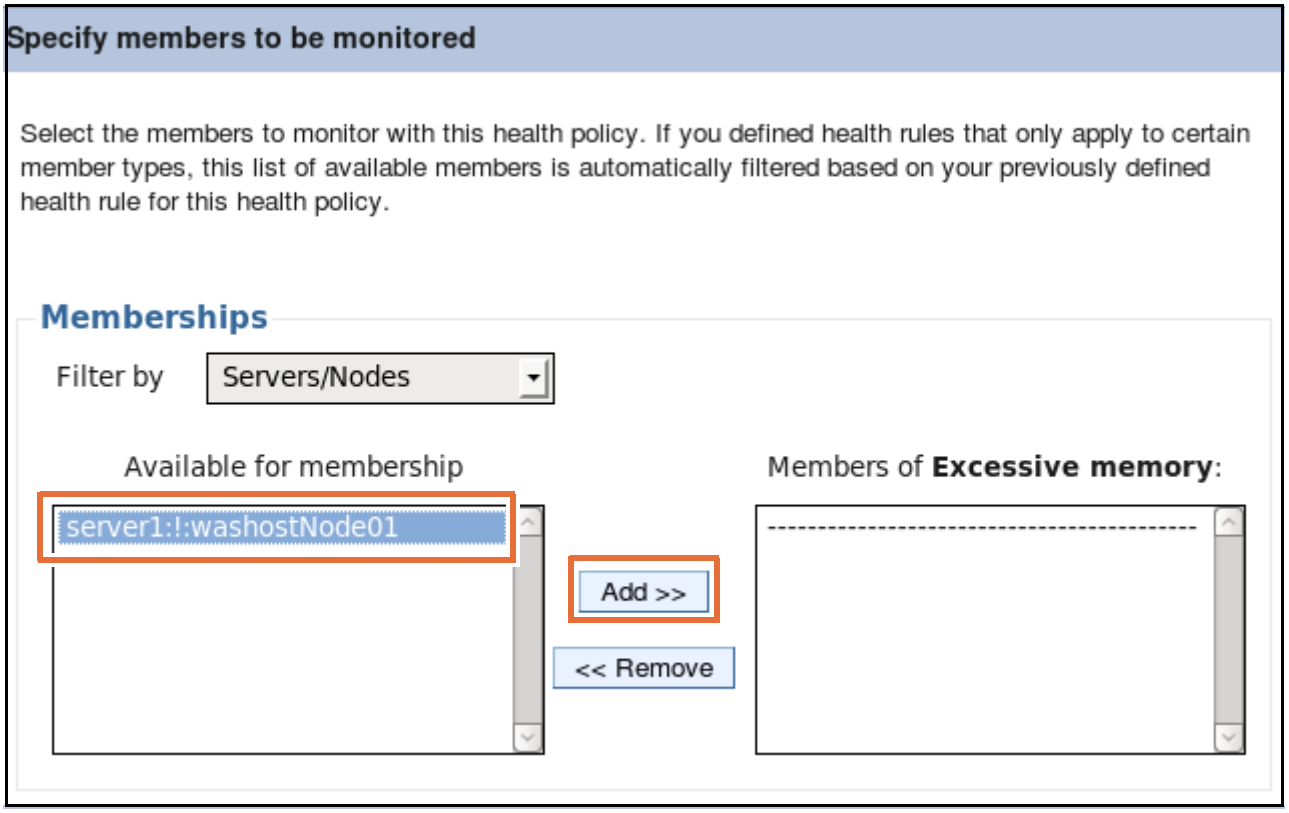
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ h. Next, you define the members to be monitored. For the Filter by value, choose **Servers/Nodes**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

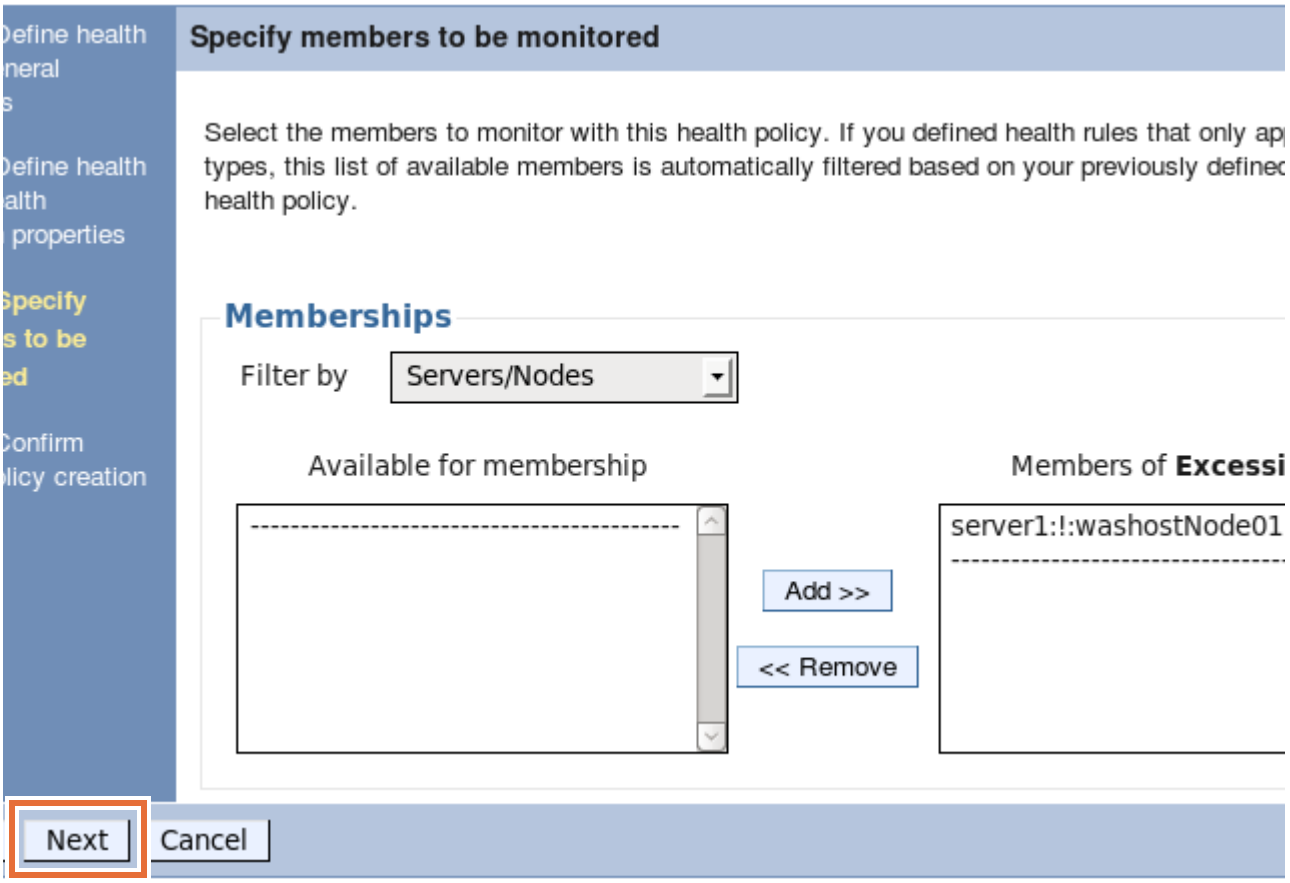
__ i. Select the **server1:!: washostNode01** server that is available for membership and click the **Add>>** option to add it to **Members of Excessive memory**.



© Copyright IBM Corp. 2013

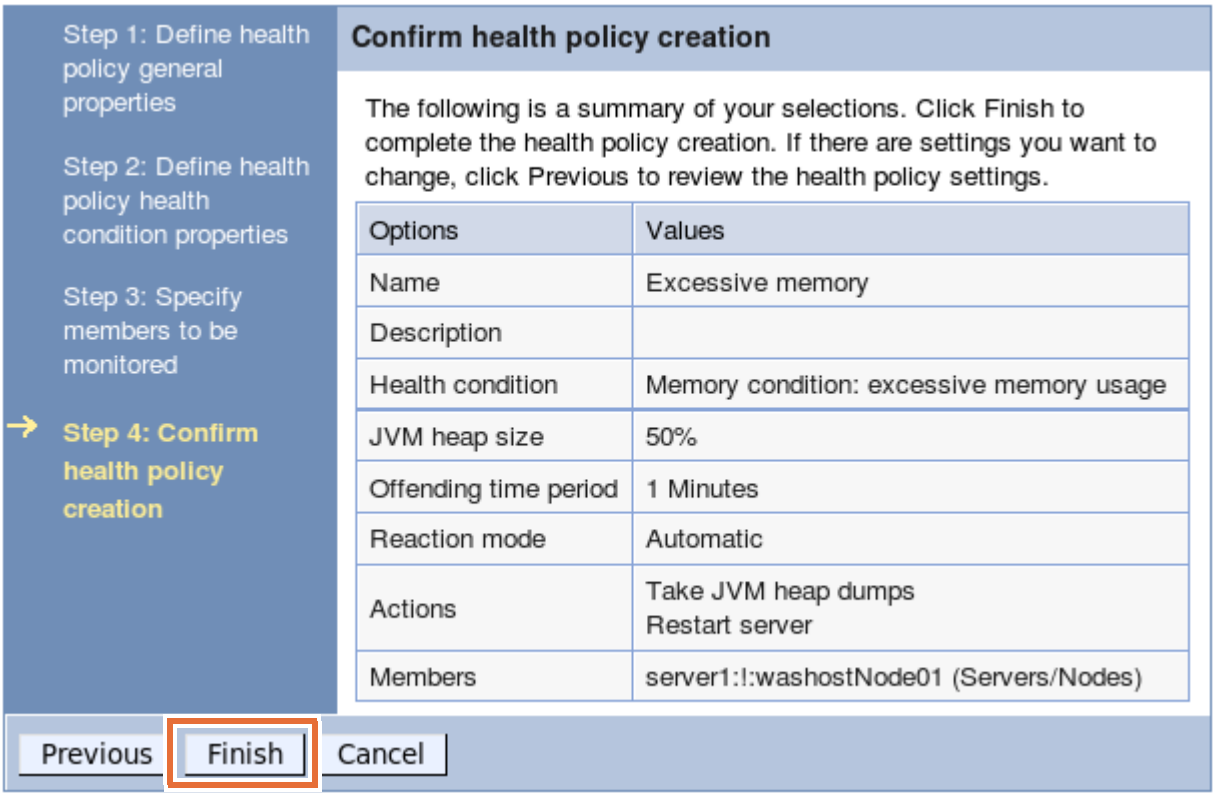
Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ j. Click **Next** to continue.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ k. Confirm the values and click **Finish**.



__ l. Click **Save** to save the changes to the repository.

__ 2. Use the Tivoli Performance viewer to monitor used memory on server1.

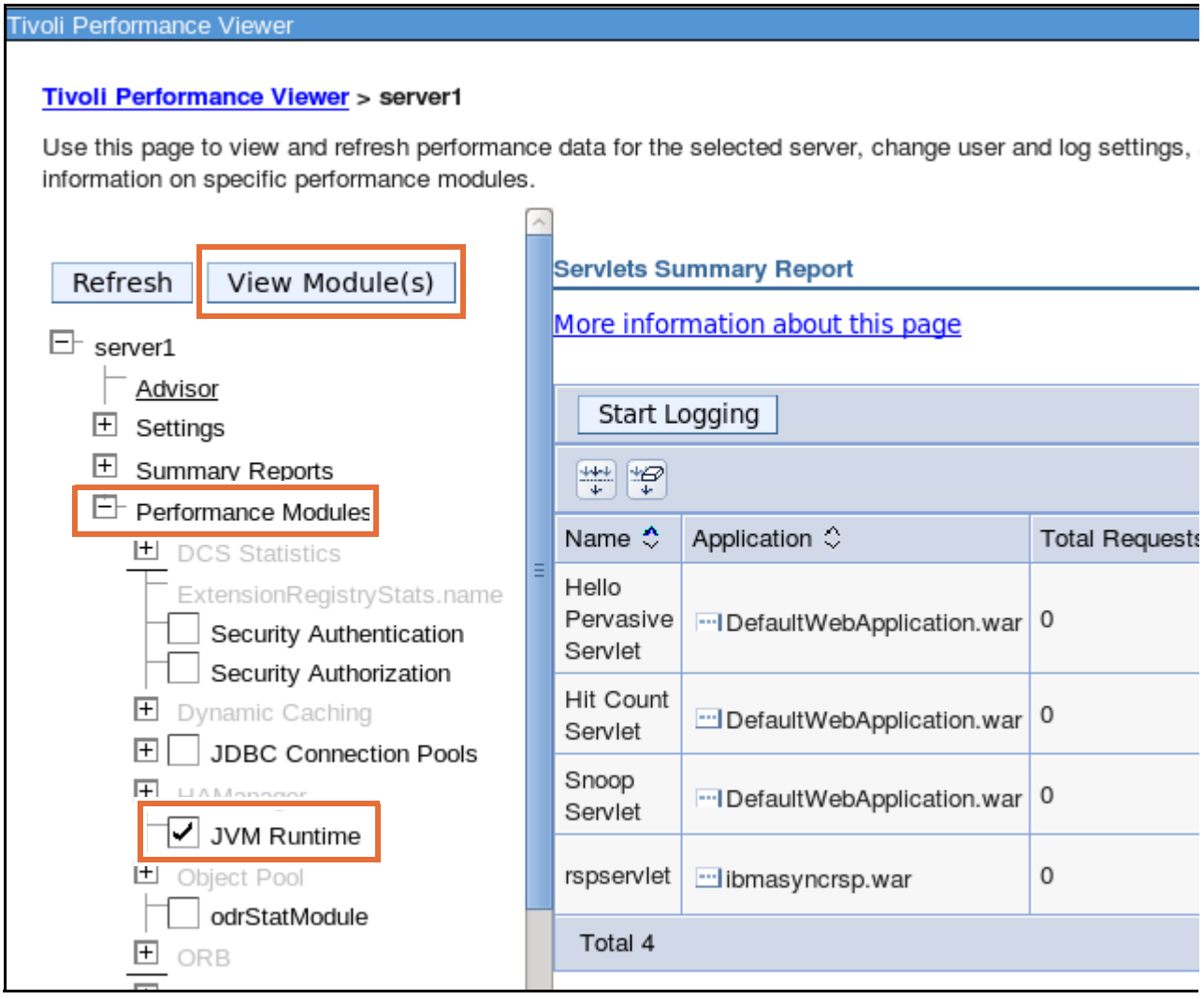
__ a. In the administration console, select **Monitoring and Tuning > Performance Viewer > Current activity**.

__ b. Click **server1**.

© Copyright IBM Corp. 2013

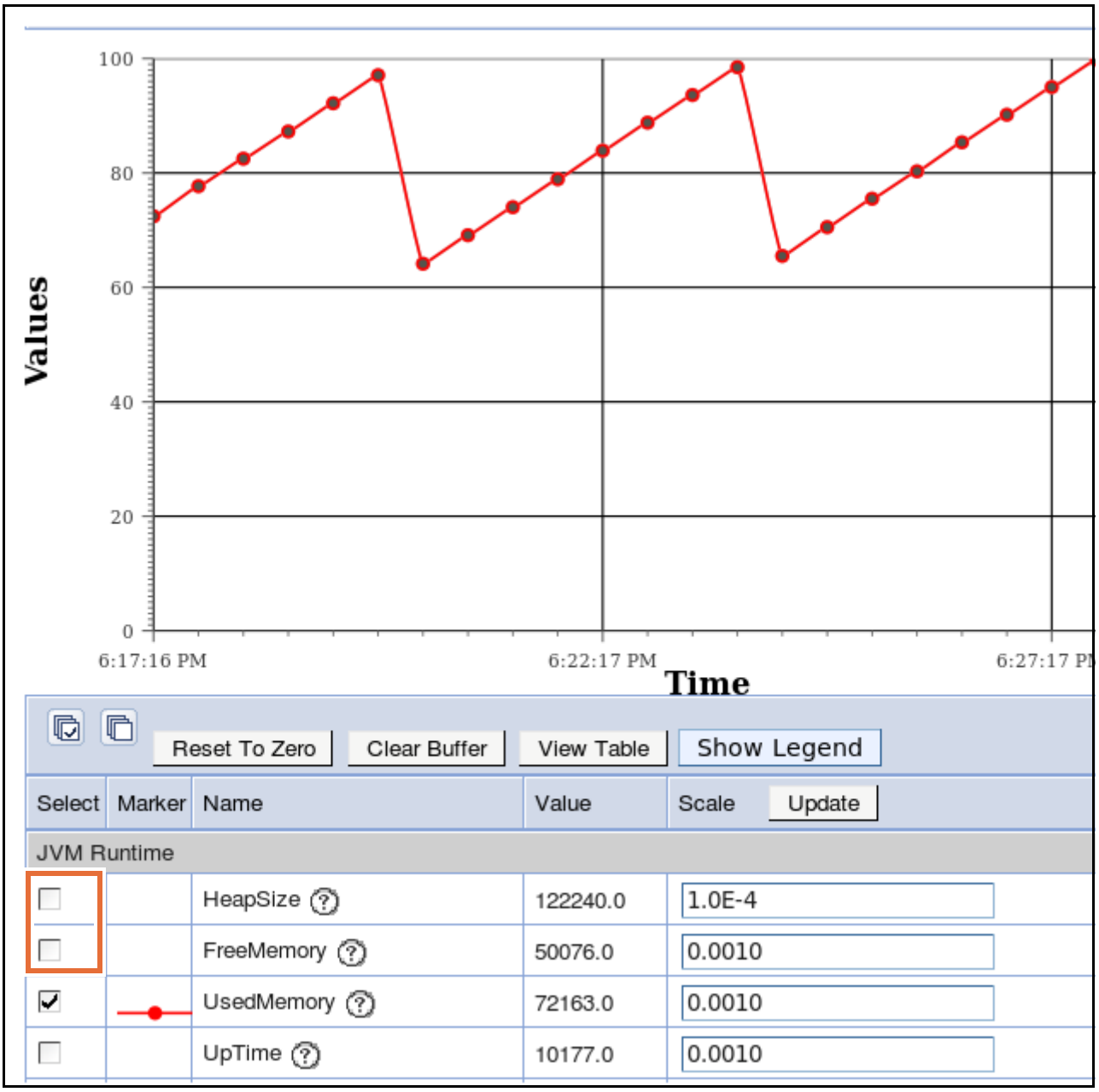
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Expand **Performance Modules**, select **JVM Runtime**, and click **View Modules**.



Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ d. Below the graph that is shown, clear the boxes for **HeapSize** and **FreeMemory** so that only used memory is displayed.



__ e. Use the resulting chart to watch the growth in used memory for server1.

Section 4: Test Health Management

__ 1. Send a request that uses excessive memory to server1.

__ a. Open a new tab in the browser. Enter the URL:

`http://localhost:9080/BadAppWebProject`

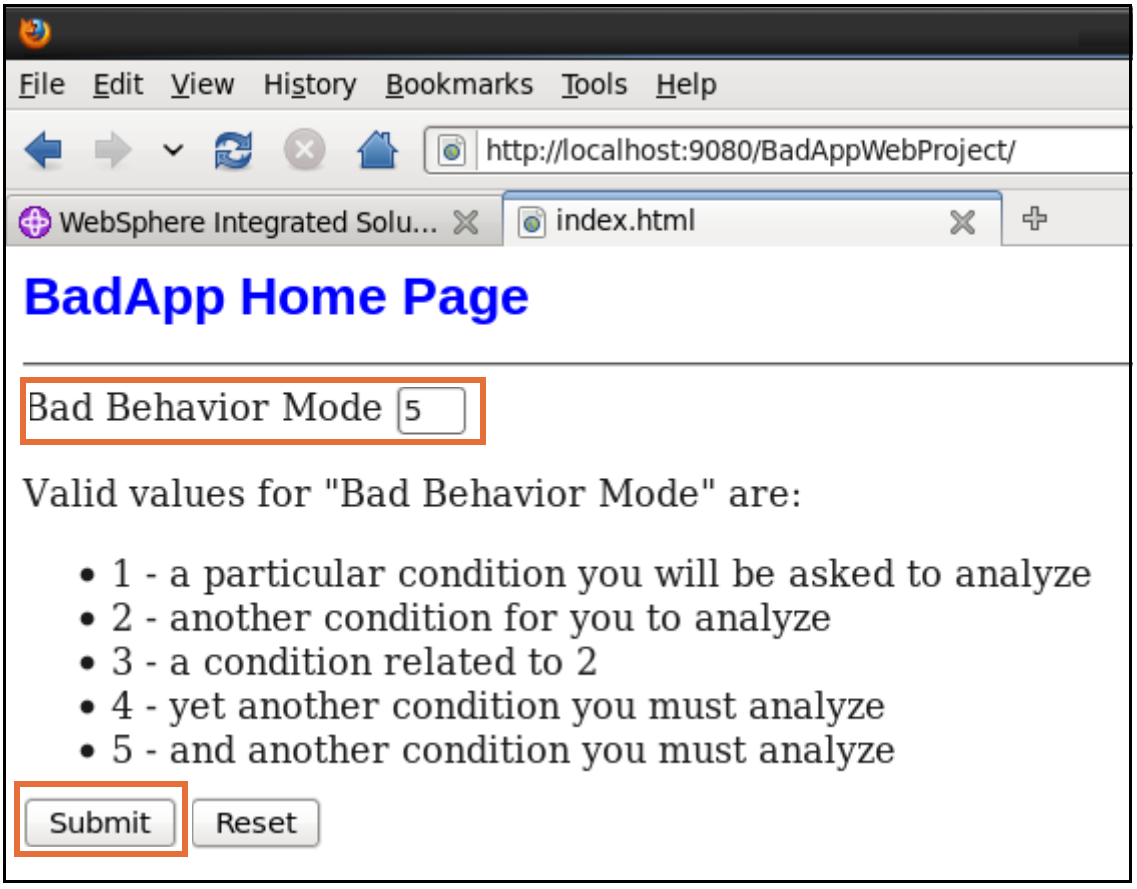
__ b. Enter **5** for **Bad Behavior Mode**.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

A Bad Behavior Mode 5 request causes the application to start allocating memory in a loop. If allowed to run long enough, the request causes an out-of-memory exception on the application server.

__ c. Click **Submit**.



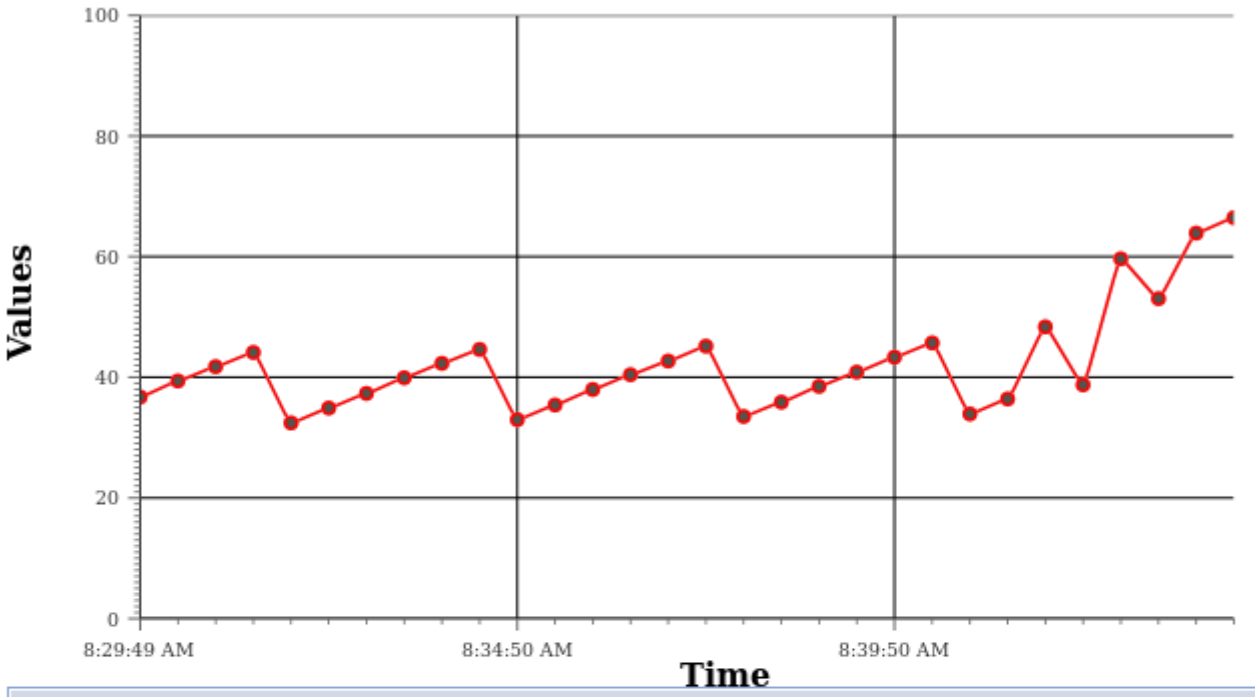
__ 2. Monitor the amount of memory that server1 uses.

__ a. Switch back to the WebSphere Integrated Solutions tab in the browser.

__ b. Using the Tivoli Performance Viewer, watch the amount of memory that server1 uses.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. Wait until the size of used memory stays above 128 MB for about a minute.



Reset To Zero | Clear Buffer | View Table | Show Legend

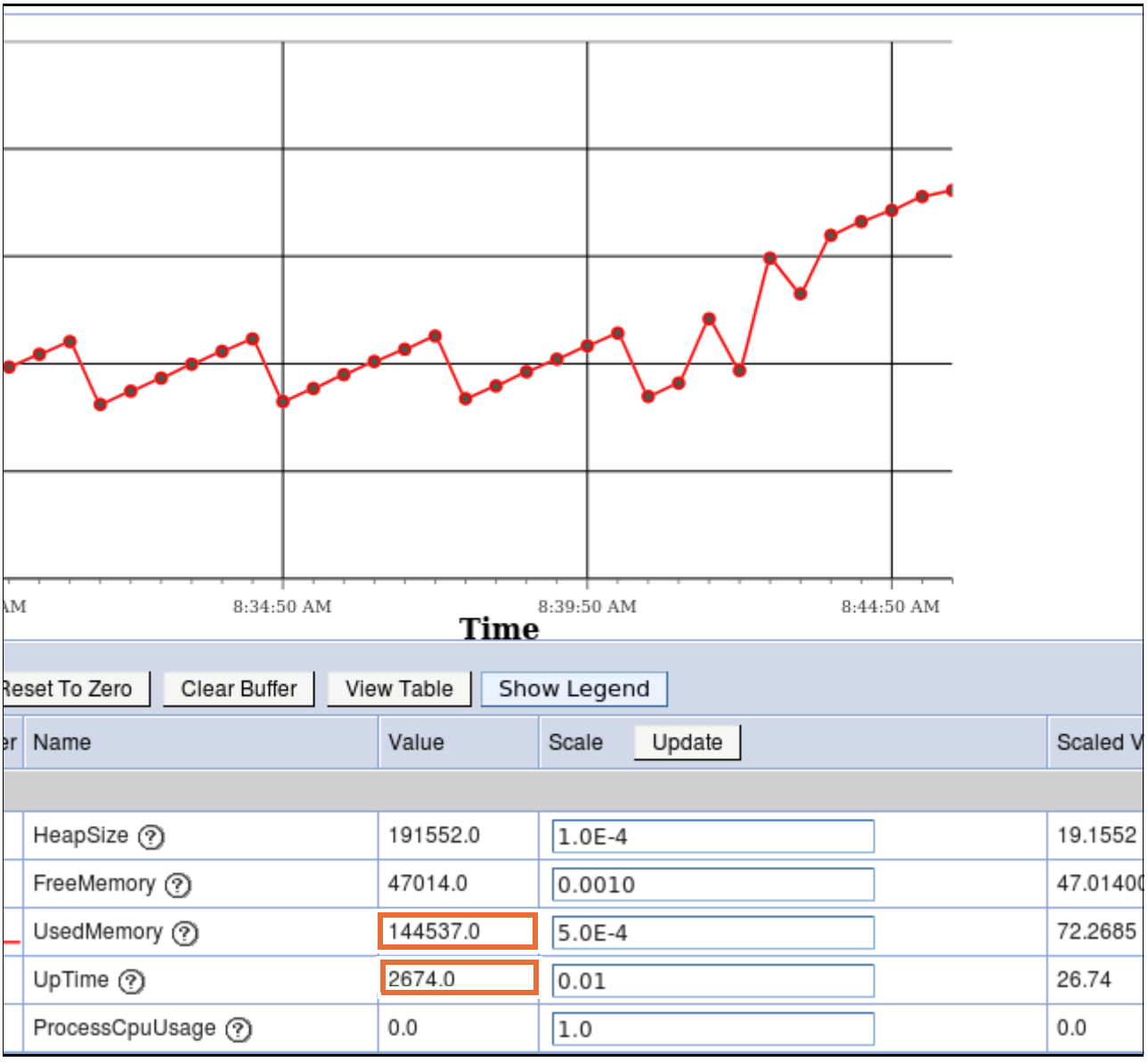
Select	Marker	Name	Value	Scale	Update
JVM Runtime					
☐		HeapSize	172800.0	1.0E-4	
☐		FreeMemory	39901.0	0.0010	
☑		UsedMemory	132898.0	5.0E-4	
☐		UpTime	2584.0	0.01	
☐		ProcessCpuUsage	0.0	1.0	

__ d. You can monitor the amount of time the heap is above 128 MB by watching the change in the UpTime value.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ e. Recall that the health controller is checking every minute. The health controller is also waiting for used memory to stay above 128 GB for more than 1 minute. It might take a few minutes for the health controller to react to the memory usage violation.



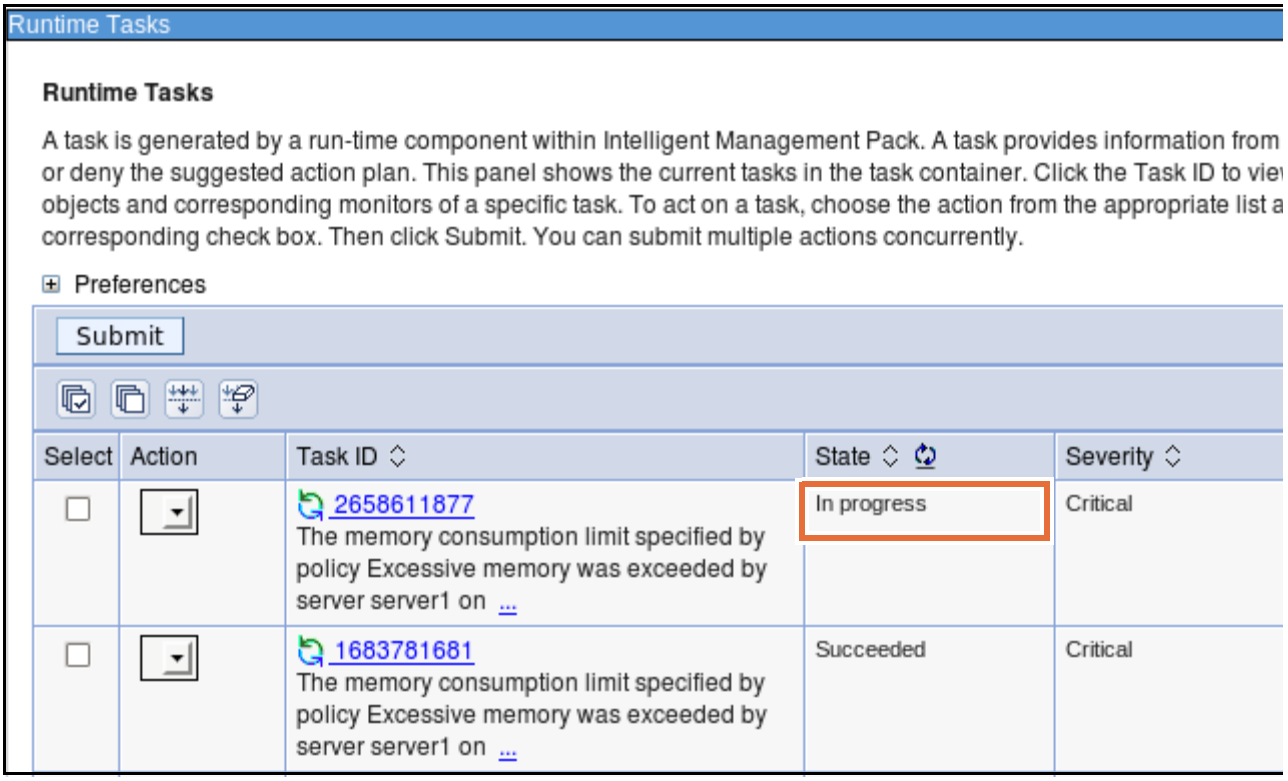
__ 3. Monitor the Health Management Controller actions.

__ a. In the administrative console, click **System administration > Task Management > Runtime Tasks**.

__ b. Note the tasks for the memory consumption violation.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ c. If there are no tasks for the memory consumption violation, wait another minute or two and select **System administration > Task Management > Runtime Tasks** again.

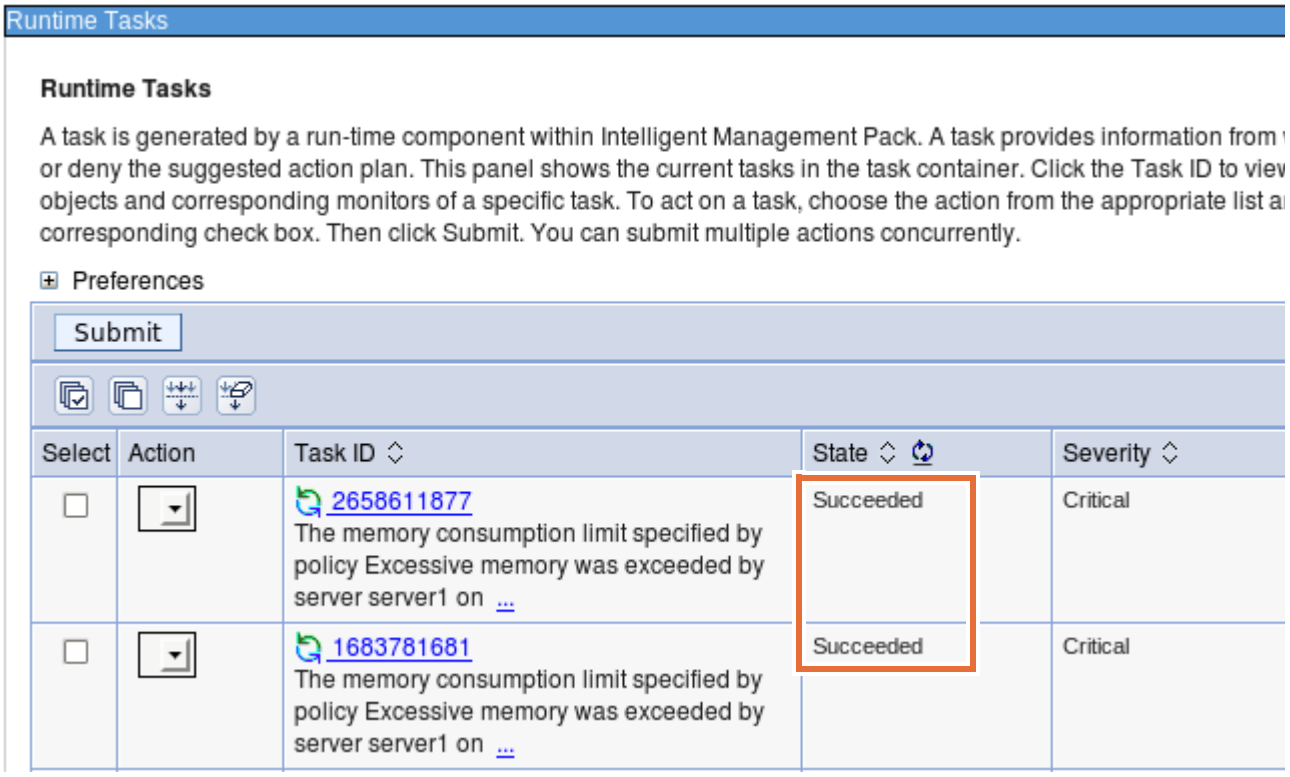


There is one task for generating the heap dump, and one task for restarting the server. The task to restart the server is **In progress**.

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ d. Refresh the task list until all tasks are in state **Succeeded**.

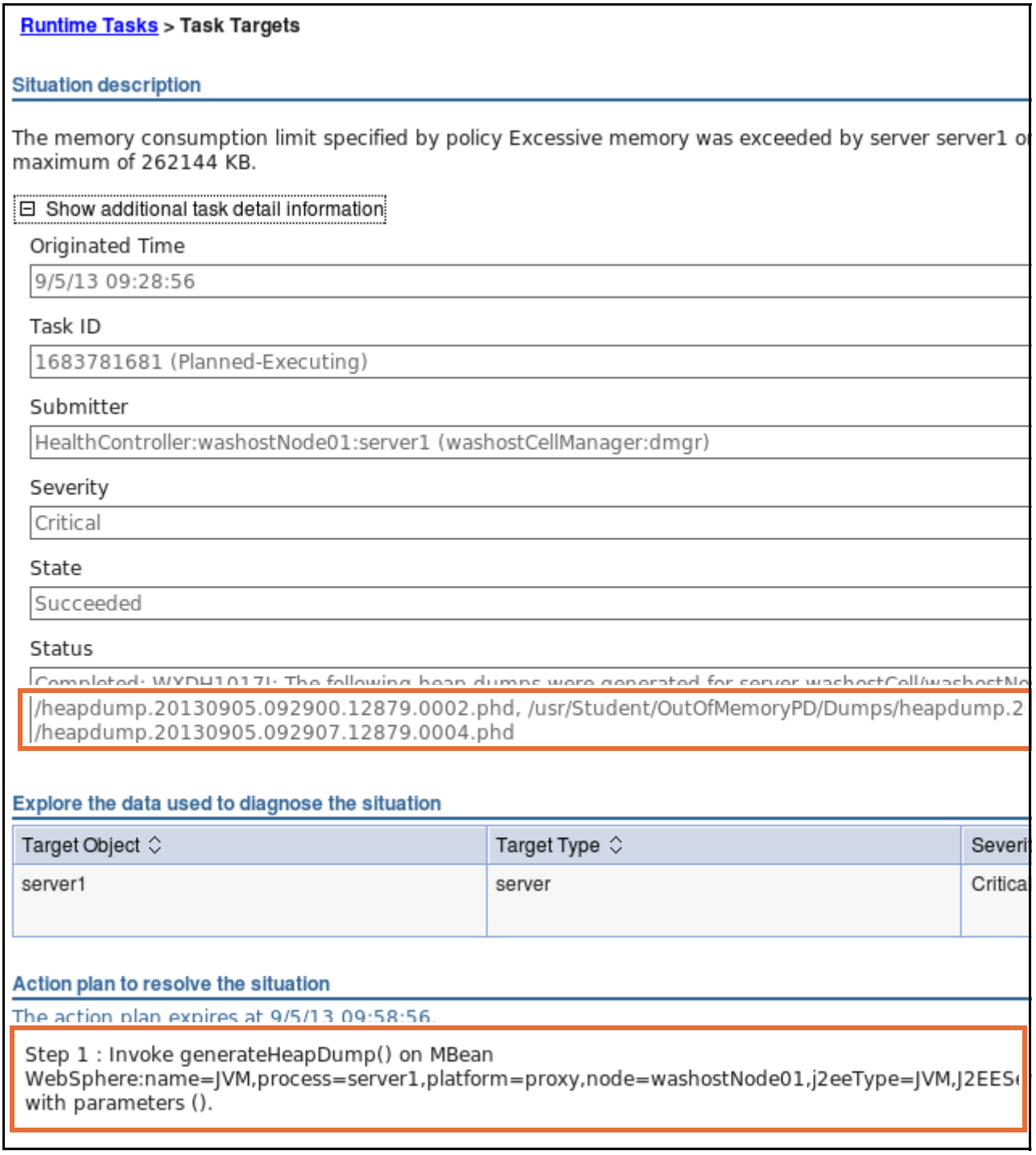


__ 4. Examine the heap dump generation task.

__ a. Click the Task ID for the task that generated the heap dump (the lower task).

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Expand **Show additional task detail information**.



__ c. Note the content of the **Status** field. The field shows what heap dumps are generated.

__ d. Note the content of the **Action plan**. The Action plan shows the command that generated the heap dump.

__ 5. Examine the restart server task.

__ a. Select **Runtime Tasks** from the breadcrumb trail.

__ b. Click the Task ID for the task that restarted the server (the top task).

 © Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

__ c. Expand **Show additional task detail information**.

Runtime Tasks > Task Targets

Situation description

The memory consumption limit specified by policy Excessive memory was exceeded by server server1 on % and the current heap size is 58 % of the maximum of 262144 KB.

⊟ Show additional task detail information

Originated Time

9/5/13 09:29:08

Task ID

2658611877 (Planned-Executing)

Submitter

HealthController:washostNode01:server1 (washostCellManager:dmgr)

Severity

Critical

State

Succeeded

Status

Completed: WXDH1008I: Server washostCell/washostNode01/server1 was restarted successfully.

Explore the data used to diagnose the situation

Target Object	Target Type	Severity
server1	server	Critical

Action plan to resolve the situation

The action plan expires at 9/5/13 09:59:08.

Step 1 : Execute the command
AdminTask.healthRestartAction(["-clusterName","null","-cellName","washostCell","-nodeName","washc

__ d. Note the content of the **Status** field. The field shows that server1 was restarted successfully.

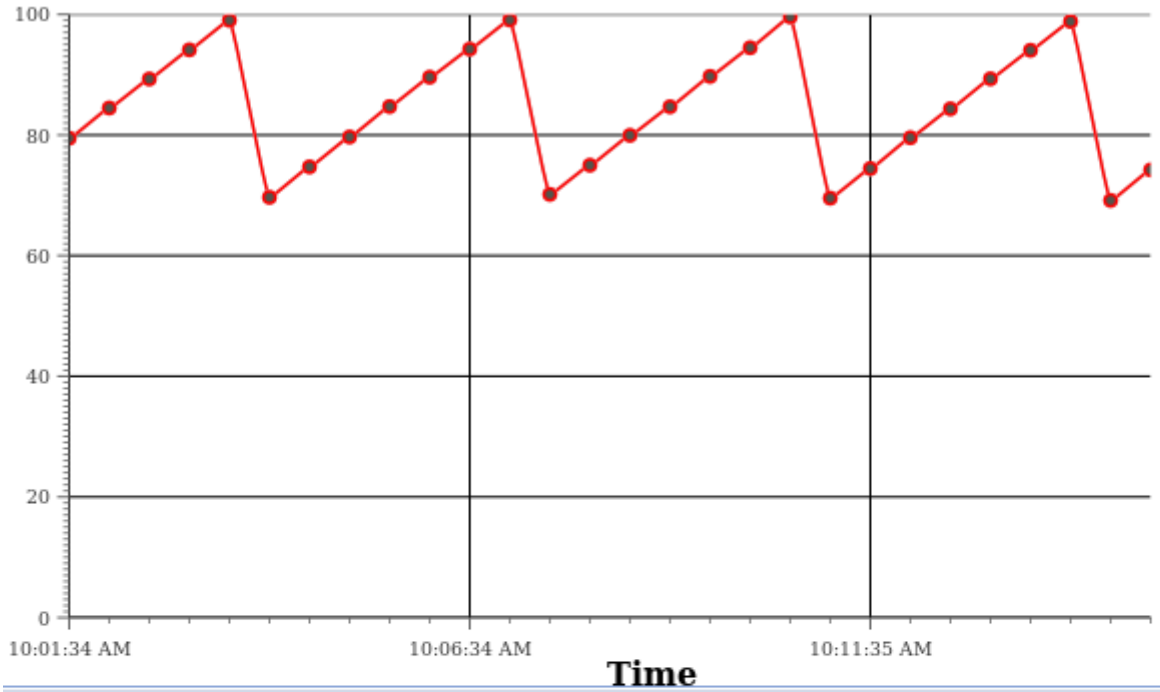
__ e. Note the content of the **Action plan**. The Action plan shows the command that restarted the server.

__ 6. Check the health of server1.

__ a. Select **Monitoring and Tuning > Performance Viewer > Current activity**.

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Click **server1**. If server1 information is not displayed, you can exit from the administrative console and log back in again.



Reset To Zero | Clear Buffer | View Table | Show Legend

t	Marker	Name	Value	Scale	Update	Scaled Value
Runtime						
		HeapSize	109056.0	1.0E-4		10.9056
		FreeMemory	34849.0	0.0010		34.849003
		UsedMemory	74206.0	0.0010		74.206
		UpTime	2508.0	0.01		25.08
		ProcessCpuUsage	0.0	1.0		0.0

__ c. Notice that **UsedMemory** is now back to normal.

__ 7. Check that the heap dumps were generated.

__ a. In a terminal window, change directory to:

`/usr/Student/OutOfMemoryPD/Dumps/`

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

__ b. Use the `ls` command to see what files are available.

```
root@washost:/usr/Student/OutOfMemoryPD/Dumps
File  Edit  View  Search  Terminal  Help
[root@washost Dumps]# cd /usr/Student/OutOfMemoryPD/Dumps/
[root@washost Dumps]# ls
heapdump.20130905.092900.12879.0002.phd
heapdump.20130905.092903.12879.0003.phd
heapdump.20130905.092907.12879.0004.phd
[root@washost Dumps]#
```

__ c. Notice that heap dumps are available for problem determination.

These heap dumps can be examined with the tools you learned about previously to track down what caused the excessive memory usage. If necessary, you can also add a generate thread dump action to the health policy action plan. When the problem is re-created, a thread dump is also available to examine.

End of exercise

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

Exercise review and wrap-up

You created a health policy to monitor the server1 application server. You used the predefined health policy memory condition: excessive memory usage. You configured the condition to trigger when 50% of the heap was in use for more than 1 minute. You created a health policy action plan that included generating heap dumps and restarting the server. You set the health policy in automatic mode so that the action plan would be performed without operator intervention.

You used the BadAppEARProject application to send a single request that would use excessive memory. The request was sent to the monitored server.

When the health controller determined that the excessive memory condition was violated, the tasks were delivered to the task management system, and the actions were run.

You examined the tasks in the task management panel, and examined the results of running the action plan. The action plan generated heap dumps and restarted the server.

© Copyright IBM Corp. 2013

Course materials may not be reproduced in whole or in part
without the prior written permission of IBM.

IBM