Practices for Lesson 8: Monitoring a Domain

Chapter 8

Practices for Lesson 8: Overview

Practices Overview

In these practices, you use the administration console to access and configure WebLogic Server log files. You also use the monitoring capabilities of the admin console and the Monitoring Dashboard to monitor various aspects of WebLogic Server.

Practice 8-1: Working with WebLogic Server Logs

Overview

In this practice, you access a WebLogic Server log file by using the administration console. You search the server log file by using an editor. You create a log filter and apply it to one of the logging outputs.

Assumptions

You completed “Practice 7-1: Configuring a JDBC Data Source.”

All servers in the domain are currently running.

Tasks

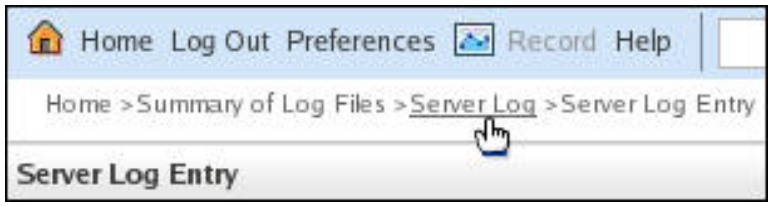
1. Connect to host01.
2. Open the WebLogic Server administration console.
3. In the admin console, locate the server log file for `server1`.
   a. In the Domain Structure, expand **Diagnostics** (click the “**+**” sign), and then select **Log Files**.
   b. In the Log Files table, select the **ServerLog** option for server1. Then click the **View** button.

   Note: If you do not see this log on the first page of the table, click the **Next** link.

   c. The Server Log Entries table displays.

   Note: There may be no rows in the table. The table settings determine what to display.

   d. Click the **Customize this table** link.
   e. Use the drop-down list to update **Time Interval**. Choose **`Last 2 day(s)`** or some other interval that will display quite a few messages. Also, use the drop-down list to change the **Number of rows displayed per page** to `25` (or a number of your choice).
   f. Click the **Apply** button.
   g. Select the option button next to a message and click the **View** button to see the details of one of the messages.
   h. Use the breadcrumbs at the top of the page to go back to the log by clicking **Server Log**.



   i. Scroll up and down and look at the log messages. Click the **Next** and **Previous** links.

   Note: You may notice that the administration console does not provide a log search capability. Because it does not, next you will open the log file in an editor.

4. In a Terminal window, navigate to the location of the log file and edit it. Use the editor to search the log file.
   a. Open a Terminal window and navigate to the `logs` directory under the `server1` directory. List the directory contents.

```
$ cd /u01/domains/part1/wlsadmin/servers/server1/logs
```

```
$ ls
access.log diagnostic_images server1.log
```

Note: There may be more files and directories in your `logs` directory than listed here.

b. Open the `server1.log` file in an editor.

```
$ gedit server1.log
```

c. Use the search capabilities of the gedit editor to find messages placed in the log by the JDBC subsystem. Press **Ctrl + F** to use the editor's search capabilities. Enter **<jdbc>** in the search field and click the **Find** button.

d. Read the first found message. Continue to click the **Find** button and look at other from the JDBC subsystem. Can you find a message that gives which JDBC driver is being used for the data source?

Hint: You can do find again, this time find: **jdbc driver**.

e. Close the Find window. Close the editor. Close the Terminal window.

5. Use the admin console to create a log filter.

Note: Here is the scenario: Let us say that you are having issues with server2 and clustering. You want to temporarily have server2 only publish messages from the Cluster subsystem into its log, so it is easier to look through the log and find the problem. You create and apply a log filter that accomplishes this.

a. In the Change Center, click **Lock & Edit**.

b. In the Domain Structure, select the domain, **wlsadmin**.

c. Click the **Configuration** tab and then the **Log Filters** subtab.

d. Above the Log Filters table, click the **New** button.

e. For the name of the filter, enter **clusterfilter**. Click **OK**.

f. In the Log Filters table, click the name of the new filter, **clusterfilter**.

g. On the **Configuration** tab of the filter, click the **Add Expressions** button.

h. Enter or select these values for the following fields, and then click **OK**:

- Message Attribute: **SUBSYSTEM**
- Operator: **LIKE**
- Value: **Cluster**

i. Click **OK**.

j. In the Change Center, click **Activate Changes**.

6. Apply the new log filter to the server log of server2.

a. In the Change Center, click **Lock & Edit**.

b. In the Domain Structure, expand **Environment** and then select **Servers**.

c. In the servers table, select **server2**.

d. Click the **Logging** tab. Ensure that the **General** subtab is selected.

e. Scroll down to find the **Advanced** link. Right above this link is the **Rotate log file on startup** check box. Select this check box to enable this feature.

Note

- In production systems, this is disabled by default. You enable it here so it is easier to see the log filter at work.
- Notice that this attribute change does not take effect until the server is restarted.

f. Click **Save**.

g. Click **Advanced**.

h. Scroll down to the “Message destination(s)” area. Under “Log file” use the **Filter** drop-down list to select `clusterfilter`.

i. Also under “Log file,” change the value of **Log File Buffer** to `2`.

Note

- This will ensure that the log file is written to more often. This is probably too small for the buffer in a real production system.
- Notice that this attribute change also does not take effect until the server is restarted.

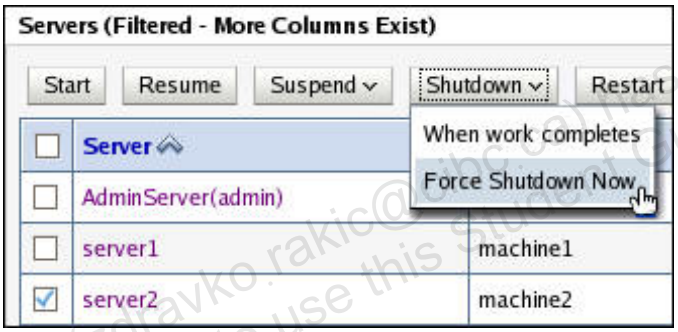
j. Click the **Save** button.

k. In the Change Center, click **Activate Changes**.

7. Check that the filter is working. First, stop and restart server2, then go look at the server2 log file.

a. In the admin console, in the Domain Structure, expand **Environment**, then select **Servers**. Click the **Control** tab.

b. Select the check box next to **server2**. Then click the **Shutdown** button and select **Force Shutdown Now**.



c. When asked to confirm the shutdown, click **Yes**.

d. Connect to host02.

e. Find the Terminal window on host02 that was running server2. You should see that server2 has shut down. Start it again by pressing the up arrow key to retrieve the command you used to previously start it, and pressing **Enter**.

Note

- Remember to enter the username and password if you do not have a boot identity file for server2.
- If you closed the previous Terminal Window, open a new window and start server2 there:

```
$ cd /u01/domains/part1/wlsadmin/bin
$ ./startManagedWebLogic.sh server2 host01.example.com:7001
```

f. When you see the message `<The server started in RUNNING mode.>`, minimize the Terminal window.

g. Open a new Terminal window and navigate to the location of server2’s log file. List the server2 log files.

```
$ cd /u01/domains/part1/wlsadmin/servers/server2/logs
$ ls server2.*
```

```
server2.log  server2.log00001
```

Note: You may have more than one older server2 log file. The newest server log file has no digits in its extension.

h. Edit the newest log file.

```
$ gedit server2.log
```

i. Look at the log file in the editor. The subsystem is after the timestamp and severity level. When server2 first comes up, the log filter has not yet been engaged. But soon it is, and from that point on, you notice that all messages are from the Cluster subsystem.

```
...
####<Feb 22, 2013 11:57:06 AM UTC> <Info> <WebLogicServer>...
####<Feb 22, 2013 11:57:27 AM UTC> <Notice> <Cluster>...
####<Feb 22, 2013 11:57:36 AM UTC> <Info> <Cluster>...
...
####<Feb 22, 2013 11:57:37 AM UTC> <Info> <Cluster>...
```

Note: The timestamps in your log file will be different. The severity levels may be different.

j. Exit the editor.

k. Close the Terminal window.

8. Remove the filter from server2.

Note: A filter like this would only be used temporarily. After you remedied whatever the cluster issue was, you would remove the filter. Without the usual messages going to the server log, it would be difficult to debug new problems as they arise.

a. Return to the web browser with the admin console.

b. In the Change Center, click **Lock & Edit**.

c. In the Domain Structure, expand **Environment** and then select **Servers**.

d. In the servers table, select **server2**.

e. Click the **Logging** tab. Ensure that the **General** subtab is selected.

f. Click **Advanced**.

g. Scroll down to the “Message destination(s)” area. Under “Log file” use the **Filter** drop-down list to select **None**.

h. Click the **Save** button.

i. In the Change Center, click **Activate Changes**.

j. If you are not proceeding to the next practice, log out of the admin console, and close the web browser.

Practice Solution: Working with WebLogic Server Logs

Perform the following tasks if you did not complete this practice and want to use the finished solution.

Assumptions

You completed “Practice 7-1: Configuring a JDBC Data Source.”

All servers in the domain are currently running.

Solution Tasks

1. Connect to host01.
2. Run the solution script.
   a. In a Terminal window, navigate to the practice directory.
      ```
      $ cd /practices/part1/practice08-01
      ```
   b. Run the solution script.
      ```
      $ ./solution.sh
      ```
      Note: This script starts the WebLogic Scripting Tool (WLST) passing it the WLST script `create_logfilter.py`, which creates the log filter. It does not apply the filter, because this filter is removed at the end of the practice. Also note that the WLST script does not change any server2 log settings that would require a server restart, nor are those settings necessary.
3. Close the Terminal window.

Practice 8-2: Monitoring WebLogic Server

Overview

In this practice, you use the monitoring capabilities of the admin console and the Monitoring Dashboard to monitor various aspects of WebLogic Server.

Assumptions

You completed “Practice 7-1: Configuring a JDBC Data Source.” (“Practice 8-1: Working with WebLogic Server Logs” is not required for this or any other practice.)

All servers in the domain are currently running.

Tasks

1. Connect to host01.
2. Access the WebLogic Server administration console.
3. Use the admin console to monitor server health.
   a. In the Domain Structure, expand **Environment** (click the “**+**” sign), and then select **Servers**.
   b. Before doing anything else, notice the Servers table. By default it displays each server’s state and health. Other columns of information can be added to the table by using the **Customize this table** link.
   c. In the Servers table, select **server2**.
   d. Click the **Monitoring** tab. Ensure that the **General** subtab is selected. Notice the information available, such as when the server started, the version of WebLogic Server, which JVM and version WebLogic Server is running under, the operating system and version, and so on.
   e. Click the **Health** subtab under the **Monitoring** tab. Some of the WebLogic Server subsystems are listed along with their health.
   f. Stay on the **Monitoring** tab of the server for the next task.
4. Modify the information displayed by a monitoring table. You will use this table later.
   a. Click the **JDBC** subtab under the server’s **Monitoring** tab.
   b. Click the **Customize this table** link. In the Chosen column, select **Active Connections Average Count** and **Active Connections High Count**. Click the left arrow ( ) to move these attributes to the Available column. In the Available column, select **Current Capacity** and **Num Available**, and click the right arrow ( ) to move them to the Chosen column. Click the **Apply** button.
5. Before monitoring a data source, check on some of its settings.
   a. In the admin console, select the data source (in the Domain Structure, expand **Services**, select **Data Sources**, and select **datasource1**).
   b. Select **Configuration** > **Connection Pool**. Make note of the Maximum Capacity.

   Note: It should be set to `5`. But if it is something different, that is OK. Just remember it.

   c. Still on this data source, click the **Monitoring** > **Statistics** tabs. Ensure that the data source is enabled and running on both servers.

   Note: If not, then perhaps one of the servers has not been started. Start it now.

   d. Minimize the web browser. You will be using it later.
6. Connect to host02. From there, run the script that sets up the database with a table and loads it with some data. Then, run the script that calls a Java client. This client uses the

data source to retrieve a connection to the database, accesses the database, and loops to do it again.

Note: Notice that this is host02, not host01.

a. Open a Terminal window and navigate to the practice directory. Set a couple of environment variables and run the script that sets up the database.

Note: The `ORACLE_HOME` variable needs to point to where the database is installed. `ORACLE_SID` (Oracle System ID) is the unique name of this particular database.

```
$ cd /practices/part1/practice08-02
$ export ORACLE_HOME=/u01/app/db11g/product/11.2.0/dbhome_1
$ export ORACLE_SID=orcl
$ ./setup.sh
...
1 row created


Commit complete.
...
$
```

Note: The SQL script drops the table (and sequence) it creates, so you can run it multiple times. Therefore, the first time the script is run, it produces messages that no such table or sequence exist. Ignore those messages if you see them.

b. Now, run the script that calls the Java client. When you run the script, you pass it the number of times you want it to access the database. The first time it accesses the database, it prints out some data it retrieves. It then loops, accessing the database, in total, the number of times you entered. The client then pauses. You let the client wait, still running, and go see what is happening with the data source.

Important: The number you enter when you run the script should be TWO TIMES the Maximum Capacity you made note of earlier. Why double it? Because the data source is targeted to the cluster, and there are two servers in the cluster, there are two instances of the data source. Therefore, the number of available connections is twice the data source connection pool **Maximum Capacity**.

Note: This Java client is written poorly on purpose. Each time the client code wants to access the database, it asks for a connection from the data source. When the code is finished with the connection, the code does not close that connection. Therefore, that connection is not returned to the connection pool.

```
$ ./runclient.sh 10
...
Your environment has been set.
>>>Obtained initial context
>>>Data source named datasource1 retrieved
>>>Querying data source...
Contacts from database:
1 Homer Simpson 742...
...
This client is still running...
When you are ready to stop it, press Enter.
```

Note: Do NOT press Enter yet!

c. Go back to the admin console on host01.

d. Navigate to the **Monitoring** > **JDBC** tab of **server2** (in the Domain Structure expand **Environment** and select **Servers**. Select **server2**. Click the **Monitoring** > **JDBC** tabs). Refresh the web browser (you can press **F5**).

e. Notice that **Active Connections Current Count** is equal to the **Maximum Capacity** you noted earlier. The **Current Capacity** value is also the **Maximum Capacity**. And **Num Available** is `0`. The bad code in the client has used up each one of the database connections.

f. In the Terminal window on host02 where the Java client is running, press the **Enter** key. The client code completes.

7. View the kinds of messages that the JDBC subsystem places in a server log.

a. Open a Terminal window and navigate to the `logs` directory under the `server2` directory. Edit the server log.

```
$ cd /u01/domains/part1/wlsadmin/servers/server2/logs
$ gedit server2.log
```

b. In the editor, scroll all the way to the bottom of the log file. Then use the search capabilities of the gedit editor to find messages placed in the log by the JDBC subsystem. Press **Ctrl + F** to use the editor's search capabilities. Because you are at the end of the file, select **Search backwards**. Enter **<jdbc>** in the search field and click the **Find** button.

c. You should see some messages that connections have been closed. This happened when the client stopped running earlier.

```
####...<JDBC>...<Connection for pool "datasource1" has been
closed.>
```

d. Close the Find window. Close the editor.

8. Use the admin console to enable JDBC debugging. This increases the number of messages about JDBC sent to the server log.

a. Go back to the admin console on host01.

b. In the Change Center, click **Lock & Edit**.

c. Select **server2**. (In the Domain Structure, click **Environment** > **Servers** and then select **server2** in the table).

d. Click the **Debug** tab.

e. In the table, expand **weblogic**. Then, select the box next to **jdbc**. Click the **Enable** button. Notice that jdbc is now "Enabled," as well as all of its children.

f. In the Change Center, click **Activate Changes**. Notice that no restarts are necessary.

9. Run the Java database client again on host02.

a. In a Terminal window on host02, once again navigate to the practice directory and run the script to call the Java database client. This time you can press Enter right away to stop the client.

```
$ cd /practices/part1/practice08-02
$ ./runclient.sh 10
...
>>>Obtained initial context
>>>Data source named datasource1 retrieved
```

```
>>>Querying data source...
Contacts from database:
1 Homer Simpson 742...
...
This client is still running...
When you are ready to stop it, press Enter.

$
```

10. View the server2 log to see the debug messages.
    a. In a Terminal window, navigate to the `logs` directory under the `server2` directory and edit the server log.
       ```
       $ cd /u01/domains/part1/wlsadmin/servers/server2/logs
       $ gedit server2.log
       ```
    b. In the editor, scroll all the way to the bottom of the log file. Then use the search capabilities of the gedit editor to find debug messages placed in the log by the JDBC subsystem. Press **Ctrl + F** to use the editor's search capabilities. Because you are at the end of the file, select **Search backwards**. Type **<debug>** in the search field and click the **Find** button.

    Note
    - Notice that there are many messages of the "debug" severity level. They are from various JDBC-related subsystems.
    - The "info" level messages you found before from the JDBC subsystem, about connections closing, are still there. In addition, there are quite a few debug messages around each one of those.
    - As you are editing this file, the editor may interrupt you with a message that the file has "changed on disk" and ask if you want to reload the file. You can reload it if you want to. Because many debug messages are being sent to the log, it is possible that the file changes as you are viewing it.

    c. See whether you can find debug messages that show connection testing. Scroll to the bottom of the file, use the find, and search backwards for the word **dual**.

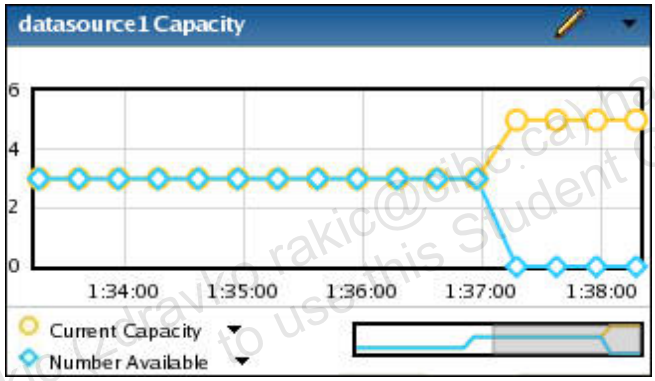
    Note: The default Test Table Name setting for a data source connection pool using an Oracle database is `SQL SELECT 1 FROM DUAL`.

    d. Close the Find window. Close the editor.
    e. Close the Terminal window.
11. Turn off debugging.
    a. Return to the admin console.
    b. In the Change Center, click **Lock & Edit**.
    c. Select **server2**. (In the Domain Structure, click **Environment** > **Servers** and then select **server2** in the table).
    d. Click the **Debug** tab.
    e. In the table, expand **weblogic**. Then, select the box next to **jdbc**. Click the **Disable** button.
    f. In the Change Center, click **Activate Changes**. Notice that no restarts are necessary.
12. Monitor a data source by using the Monitoring Dashboard. You will run the database client one more time.

a. Go to the home page of the admin console. (You can use the Home link.)
b. Under “Charts and Graphs,” click the **Monitoring Dashboard** link. The dashboard opens in a new window (or tab).
c. In the Monitoring Dashboard View List, expand **server2**. Select the view **JDBC Data Sources on server2**.
d. Above the view list, press the Start button ( ).
e. In a Terminal window on host02, navigate to the practice directory and run the script to call the Java database client. When the client is waiting, do *not* press Enter, let it continue to run.

```
$ cd /practices/part1/practice08-02
$ ./runclient.sh 10
...
This client is still running...
When you are ready to stop it, press Enter.
```

f. Return to the Monitoring Dashboard. You should notice changes in the “datasource1 Capacity” chart (top left). Your chart may not look exactly like this one, but the “Number Available” should go to 0.



g. In the “datasource1 Connections” chart, you should also notice that the “Active Connections Current Count” goes up.
h. Above the View List, click the Stop button ( ).
i. Close the Monitoring Dashboard window (or tab). When asked whether you are sure, click the **Leave Page** button.
j. Log out of the admin console.
k. Close the web browser.
l. In the Terminal window on host02 that is running the Java client, press Enter. Once the client stops, close the Terminal window.

Practice Solution: Monitoring WebLogic Server

There is no solution for this practice.