

## IBM WEBSHERE WORKSHOP - LAB EXERCISE

# WebSphere 4.0 - Performance

### **What This Exercise is About**

In this exercise you will look at some of the new performance features and tools available in WebSphere 4.0. You will set instrumentation levels on specific modules with the Admin Console and then use the Resource Analyzer and the Performance Servlet to view performance data on those modules in XML, an Open Standard. Finally, you will also use the Performance Tuner to change different settings through the GUI wizard.

### **User Requirement**

For this lab, you will need the following software:

- WebSphere 4.0.1 Advanced Edition (Driver a0131.07)
- DB2 v7.2a
- Netscape v4.7

This lab requires one system with Windows NT 4.0 Server with Service Pack 6. IBM DB2 Universal Database Version 7 with FixPack 2a is required. This lab is based on WebSphere Application Server V4.0.1 Advanced Edition driver a0131.07. This lab requires the MyBank application components installed from the previous lab. The lab also requires the PerfServletApp.ear file, shipped with WebSphere, for completion

### **What You Should Be Able to Do**

Upon completing this lab, you will know how to set the instrumentation level for collecting performance data and obtaining the data for display using the Resource Analyzer. You will also know how to install the Performance Servlet and make a request to the servlet returning collected data for display in a web browser. At the end of this lab, you will also know how to use the Performance Tuner to change different performance settings through the GUI wizard interface.

### **Introduction**

You will use the WebSphere Advanced Admin Console to set the instrumentation levels to monitor the JVM and different EJB values. With the levels of collection set, the Resource Analyzer will be used to view the data. Next, you will install the Performance Servlet into the node. After viewing the data obtained by the Performance Servlet, you will use the Performance Tuner to change the settings with in your application.

### **Exercise Instructions**

The lab consists of the following parts:

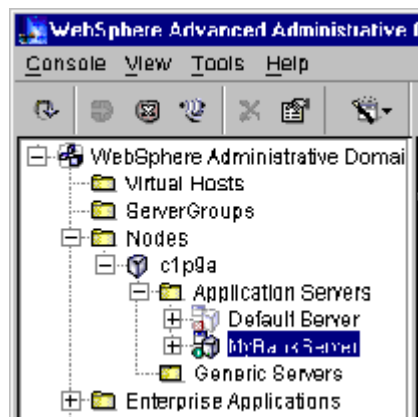
Part One: Set Instrumentation Level from Admin Console

Part Two: Using the Resource Analyzer  
Part Three: Using the Performance Servlet  
Part Four: Using the Performance Tuner

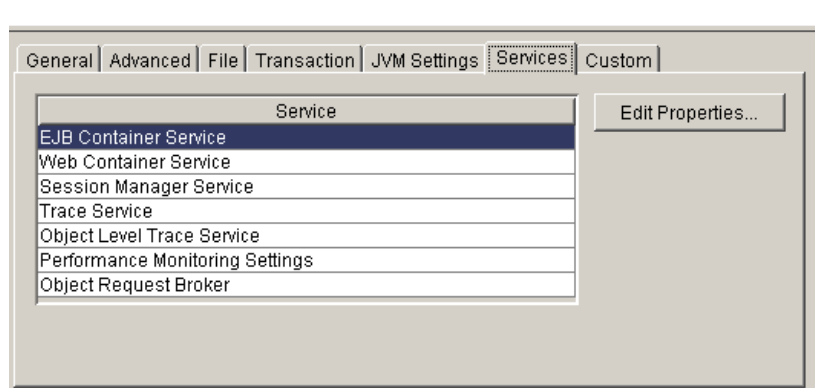
The MyBank application will need to be installed in WebSphere and fully functional.

### **Part One: Set Instrumentation Level**

- \_\_1. Start the WebSphere 4.0 Advanced Edition server from a command line (if not started):  
**adminserver**
  
- \_\_2. Start the WebSphere Advanced Administrator Console from a command line:  
**adminclient**  
  
\_\_ If security is still enabled, login with the appropriate userid and password (db2admin and was1edu).  
\_\_ Start the **MyBankServer** if it is not started.
  
- \_\_3. When the console comes up, expand **WebSphere Administrative Domain**, **Nodes**, **<computer name>**, and **Application Servers**. Select **MyBankServer**.

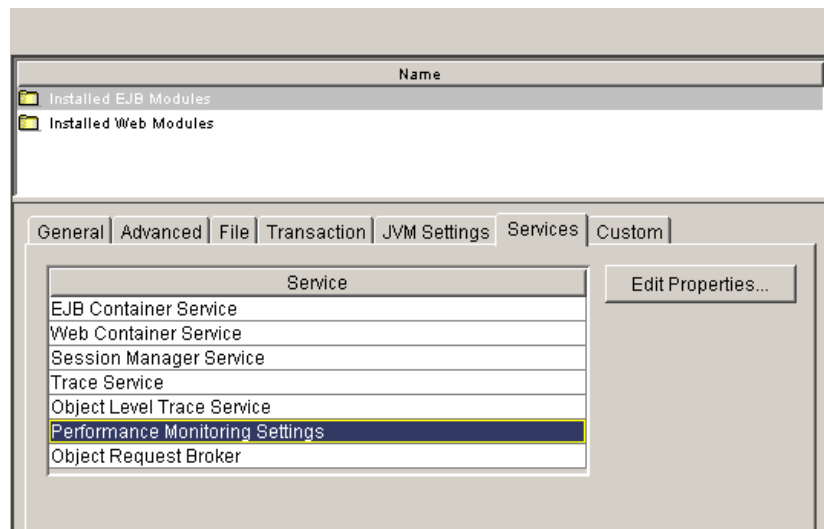


- \_\_4. On the right side of the console, select the **Services** tab for **MyBankServer**.

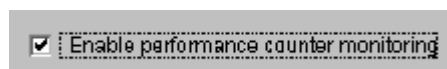


- \_\_5. From the **Service** table, select **Performance Monitoring Settings**.

- \_\_6. Select the **Edit Properties...** button.

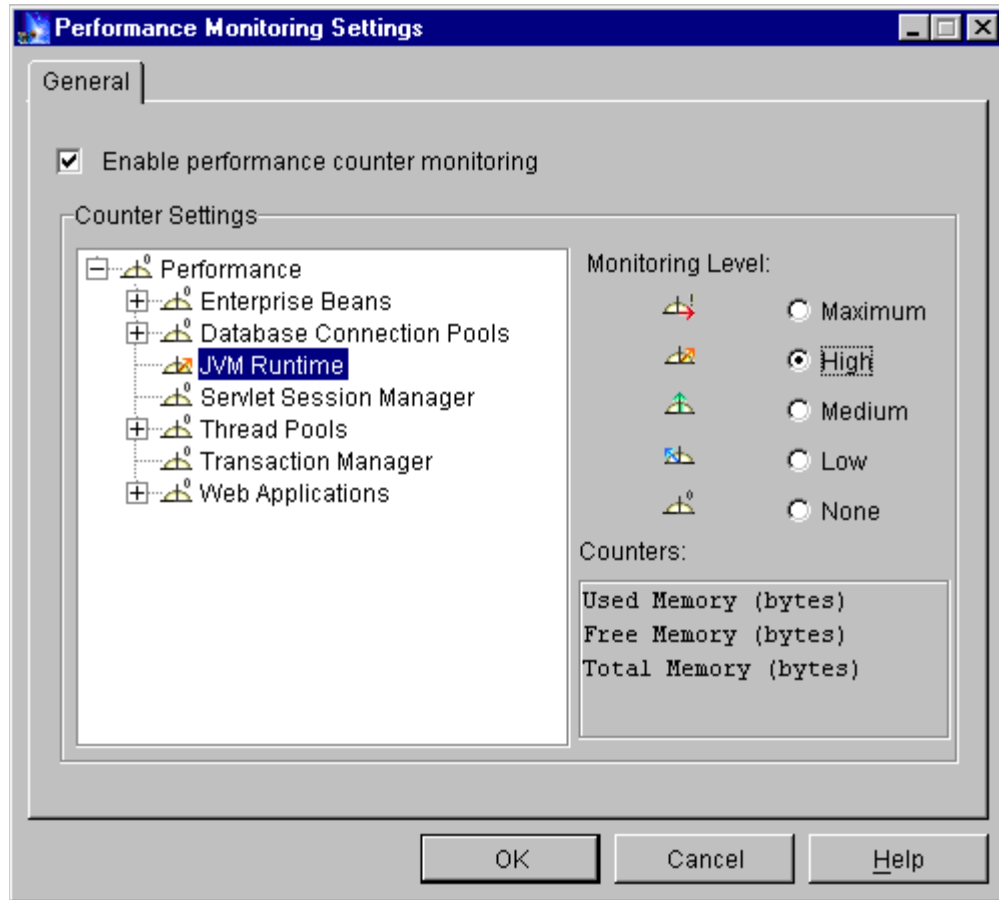


- \_\_7. In the Performance Monitor Settings window, check the box **Enable performance counter monitoring**.



- \_\_8. Expand **Performance** in the Counter Settings section.

\_\_9. Select **JVM Runtime** and select the **High** radio button.

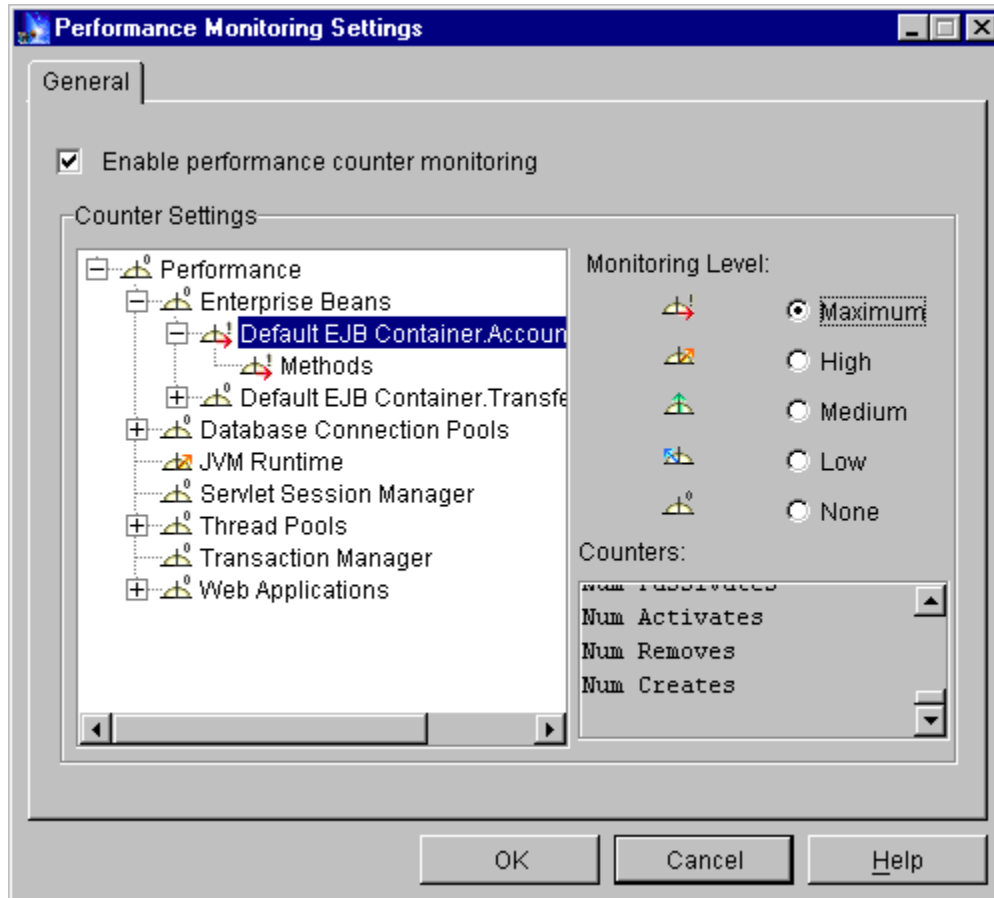


\_\_10. Next expand the **Enterprise Beans** item.

\_\_11. Expand **Default EJB Container.AccountHomeGlobal**.

\_\_ Select **Default EJB Container.AccountHomeGlobal**.

\_\_ Select the **Maximum** radio button on the right.



\_\_12. Select **OK**.

\_\_13. In the WebSphere Advanced Admin Console, select the **Apply** button.

## Part Two: Using the Resource Analyzer

\_\_14. From the WebSphere Advanced Admin Console, select **Resource Analyzer** from the **Tools** menu in the WebSphere Advanced Admin Console.

\_\_ If security is enabled, enter the userid and password used to secure the console (db2admin and was1edu).

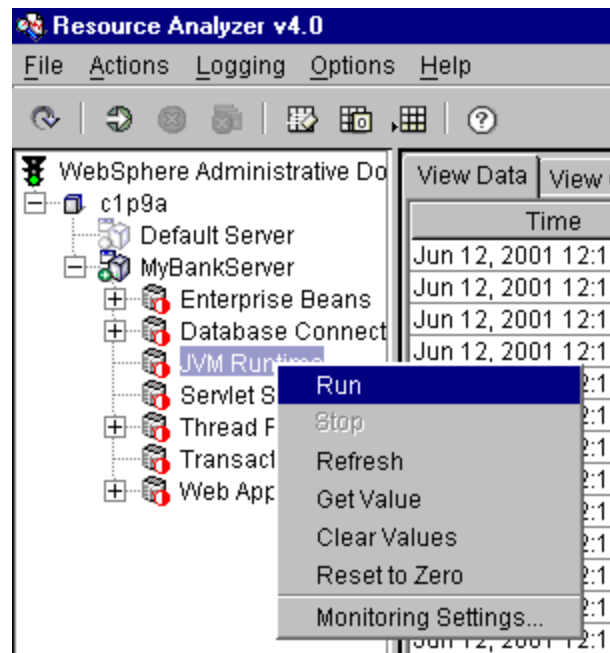
\_\_2. Even though the modules to monitor are turned on, we have to start collecting data using the Resource Analyzer.

\_\_ After the Resource Analyzer starts, expand the **<hostname>**.

\_\_ Expand **MyBankServer**.

\_\_ Select **JVM Runtime**.

\_\_ Right-click and select **Start**.



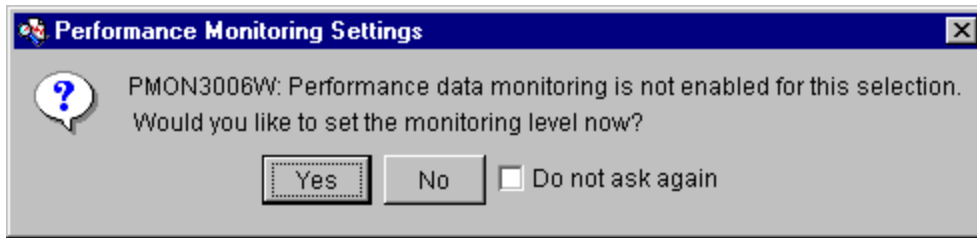
\_\_ Wait a few seconds for rows to be added to the table. Select the **View Chart** tab to see a graphical change of the JVM statistics.

- \_\_\_ 3. Besides the JVM statistics, monitors for the EJBs were also set. Statistics can be viewed on the entity bean.
  - \_\_\_ Expand Enterprise Beans in the left-pane navigator
  - \_\_\_ Select **Default EJB Container.AccountHomeGlobal** (You may have to use the scroll bar to view the complete module name).
  - \_\_\_ In the table in the lower right portion of the window, check the boxes **Total Method Calls**, **Avg Method RT (ms)**, and **Avg Create Time (ms)**.
  - \_\_\_ Remove the check from the **Num Removes** box.
  - \_\_\_ Select the **Start** button from the toolbar.

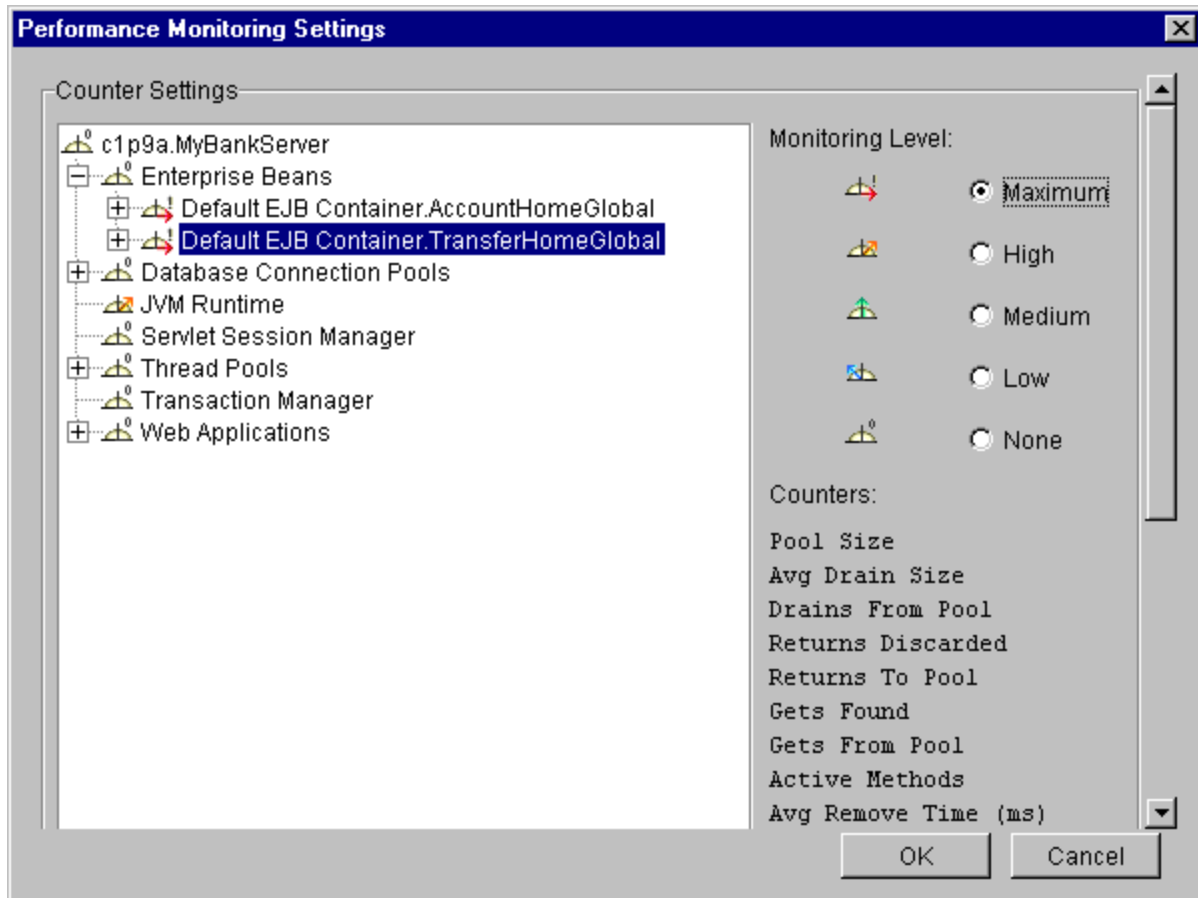


Notice the statistics begin to be listed.

- \_\_\_ 4. Statistics can also be collected and viewed for the Transfer session bean. First the instrumentation level for the Transfer session bean must be set before statistics can be run.
  - \_\_\_ Select **Default EJB Container.TransferHomeGlobal** (You may have to use the scroll bar to view the complete module name).
  - \_\_\_ Select **Yes** to the message about turning on the monitors.



- \_\_\_ In the Performance Monitoring Settings window, expand **Enterprise Beans**.
- \_\_\_ Select **Default EJB Container.TransferHomeGlobal** if it is not selected.
- \_\_\_ Select the **Maximum** radio button on the right.
- \_\_\_ Select **OK**.



- In the table in the lower right portion of the window, check the boxes **Num Instantiates**, **Total Method Calls**, **Avg Method RT**, and **Avg Create Time**.
- Remove the checks from **Num Creates**, **Num Removes**, and **Num Activates**.

5. Open Internet Explorer and load the MyBank URL:

**<http://localhost:9081/MyBank/index.html>**

6. Complete a number of transactions to create statistics (provide the correct userid and password when prompted).

- Create a savings account number 10 with a balance of 1000
- Create a checking account number 11 with a balance of 100
- Transfer 50 from account 10 to account 11
- Transfer 40 from account 10 to account 11
- Transfer 30 from account 10 to account 11
- Transfer 25 from account 10 to account 11

7. Return to the Resource Analyzer and view the Data and Chart tabs for the Account EJB. Notice the number of creates matches the number of accounts created. Also look at the Avg Create Time which is in microseconds.

8. Now look at the Data and Chart tabs for the Transfer session bean. Notice the bean has been instantiated only once. Also note the Avg Create Time.

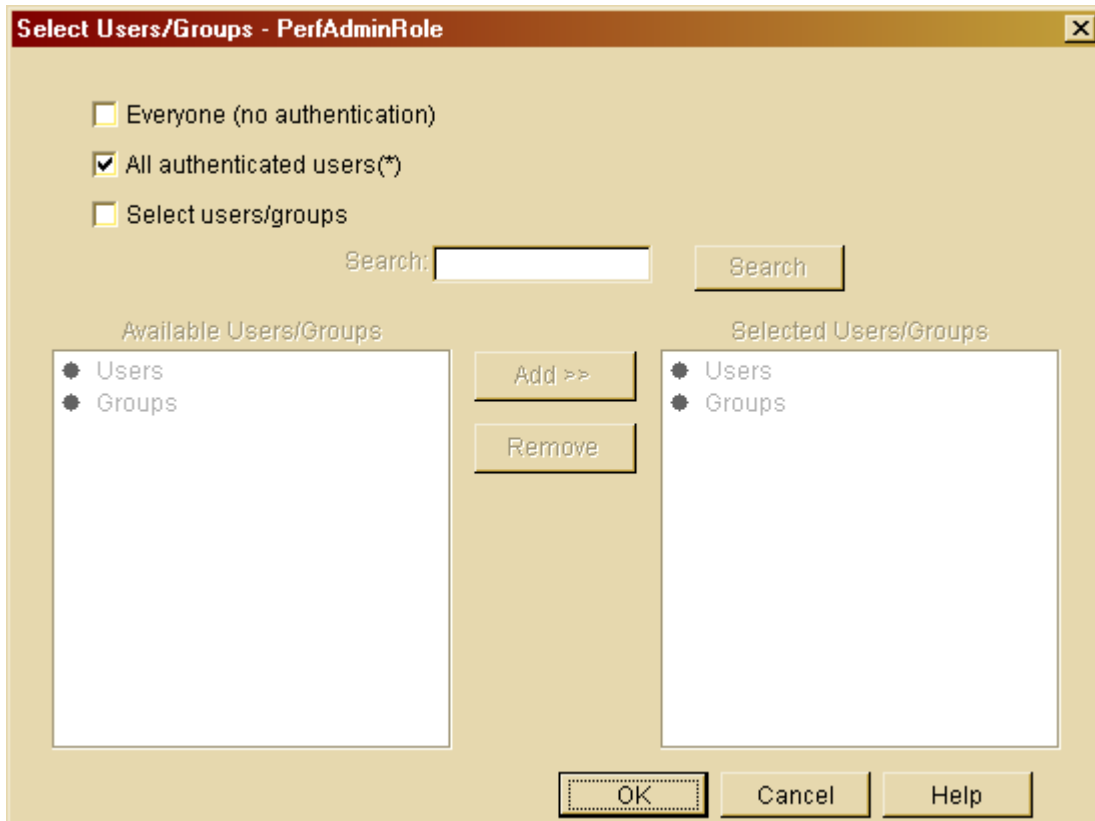
9. Return to the Chart of the JVM Runtime. Notice that the Free Memory is decreasing as the EJBs are created.

### **Part Three: Using the Performance Servlet**

The Performance Servlet is an interface to PMI allowing for performance data to be accessed over HTTP and formatted in XML. By using an Open Standard format of XML, performance data is easy to access and use in other applications. Using the Instrumentation settings from Part One and Part Two, we will access the same performance data we viewed with the Resource Analyzer.

- \_\_1. Before obtaining data from the Performance Servlet, you must install it into an application server. In the Admin Console, select the Wizard icon and select **Install Enterprise Application**.
  
- \_\_2. Verify the **Install Application (\*.ear)** radio button is selected.
  - \_\_ Click **Browse** and navigate to **c:\Websphere\AdvancedEdition\InstallableApps** and select **perfServletApp.ear**.
  - \_\_ Click **Open**.
  - \_\_ Enter Application Name as **PerfServletApp**.
  
  - \_\_ Click **Next** twice until the **Mapping Users to Roles** window is displayed.
  - \_\_ Select the Role **PerfAdminRole**.
  - \_\_ Click the **Select...** button.

\_\_ In the **Select Users/Groups - PerfAdminRole** window click the box **All authenticated users(\*)**.



\_\_ Select **OK**.

You have just changed the access to the performance servlet without using the assembly tool.

\_\_ Continue to click **Next** until the **Selecting the Application Server** window is displayed.

\_\_ Click on **Select Server** button.

\_\_ Select **MyBankServer(<node name>)** and click **OK**. Click **Next**.

\_\_ Click **Finish**.

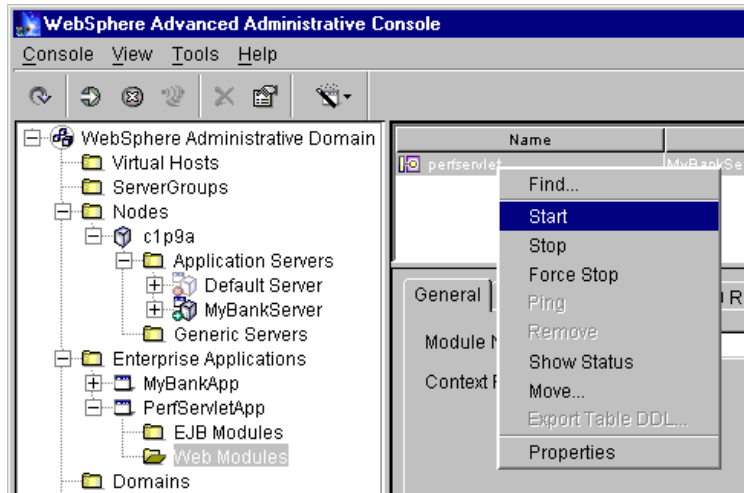
\_\_3. Once the informational box appears for Enterprise Application Install Completed Successfully, click **OK**

\_\_4. If the MyBankServer is still running we will simply need to start the **perf servlet** in the **Web Module** of the **PerfServletApp** Enterprise Application.

\_\_ In the Admin Console, expand **Enterprise Applications**.

\_\_ Expand **PerfServletApp** and **Web Modules**.

\_\_ In the right hand view, right-click **perfServlet** and select **Start**.



\_\_ When the application starts, select **OK** to close the Information dialog.

\_\_5. Using Internet Explorer, specify the following URL:

**<http://localhost:9081/wasPerfTool/servlet/perfservlet>**

You will see a list in XML of the different application servers and data from modules with instrumentation levels set from Part One and Part Two.

\_\_7. Complete a number of transactions to create more statistics.

- \_\_ Access the MyBank application again from a **new** web browser (<http://localhost:9081/MyBank/index.html>.)
- \_\_ Create a savings account number 12 with a balance of 2000
- \_\_ Create a checking account number 13 with a balance of 2000
- \_\_ Transfer 150 from account 12 to account 13
- \_\_ Transfer 420 from account 12 to account 13
- \_\_ Transfer 310 from account 12 to account 13
- \_\_ Transfer 125 from account 12 to account 13

\_\_8. Go back to the browser with the Performance Servlet results from before and refresh the page.

**<http://localhost:9081/wasPerfTool/servlet/perfservlet>**

Notice the change in the creates (2) for the AccountHome EJBModule.  
Also notice the number of instantiates for the TransferHome EJBModule.

- \_\_9. Data can be obtained from specific modules by specifying arguments to the Performance Servlet. Specify the Node, Server and beanmodule as arguments to only view the EJB modules.

<http://localhost:9081/wasPerfTool/servlet/perfservlet?Node=<hostname>&Server=MyBankServer&Module=beanModule>

**\*\*\*NOTE\*\*\*:** Make sure to substitute the hostname of the computer into the correct place in the URL.  
The URL is case-sensitive.

- \_\_10. To view only the JVM information, specify the following URL.

<http://localhost:9081/wasPerfTool/servlet/perfservlet?Node=<hostname>&Server=MyBankServer&Module=jvmRuntimeModule>

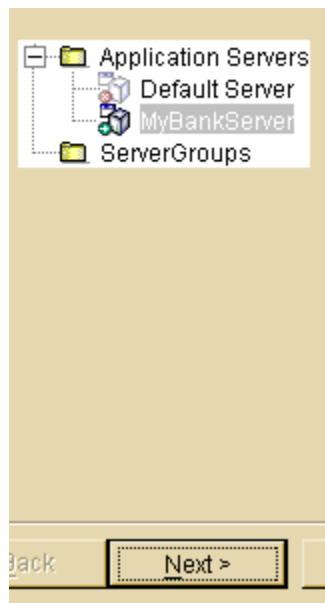
The following information should be displayed.

```
<PerformanceMonitor>
  <Node name="c1p9a">
    <Server name="MyBankServer">
      <jvmRuntimeModule>
        <totalMemory>
          <PerfNumericInfo time="992392873421" uid="pmi1"
                                val="6.7107832E7" />
        </totalMemory>
        <freeMemory>
          <PerfNumericInfo time="992392873421" uid="pmi2"
                                val="2.5779768E7" />
        </freeMemory>
        <usedMemory>
          <PerfNumericInfo time="992392873421" uid="pmi3"
                                val="4.1328064E7" />
        </usedMemory>
      </jvmRuntimeModule>
    </Server>
  </Node>
</PerformanceMonitor>
```

Hit the refresh button and watch the performance data change.

## **Part Four: Using the Performance Tuner**

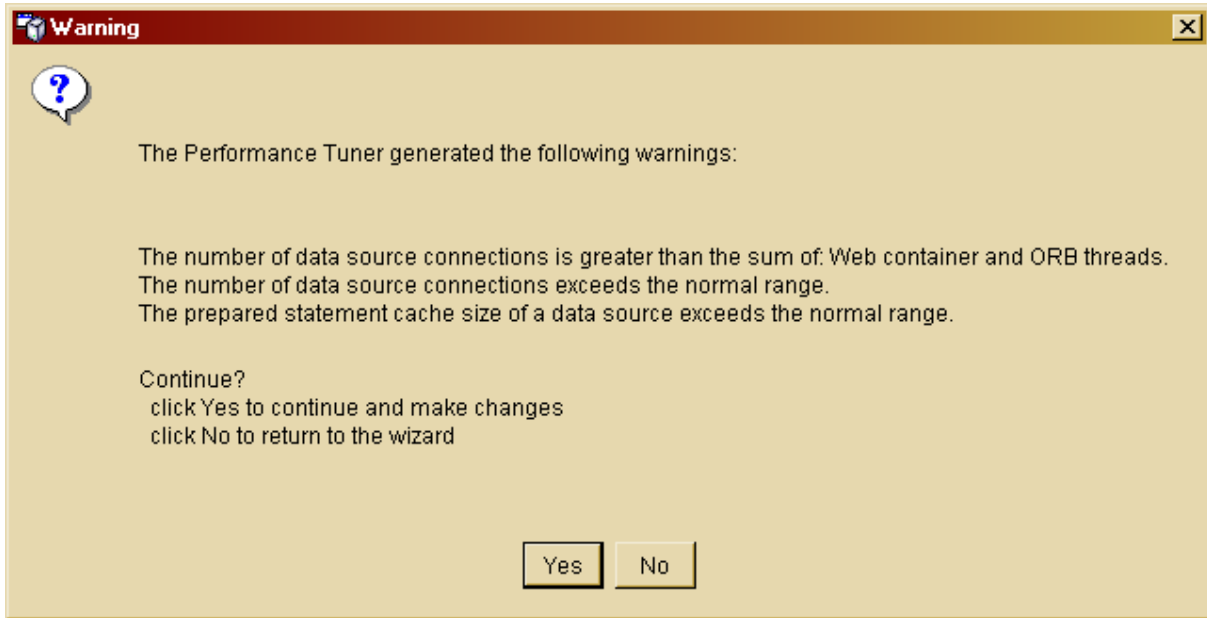
- \_\_1. From the **WebSphere Advanced Admin Console**, select the **Console** menu and then Wizards and Performance Tuner.
- \_\_2. In the Performance Tuner: Welcome window, select **Next**.
- \_\_3. In the **Application Server** window, select the **MyBankServer** before selecting **Next**.



- \_\_4. In the Performance Tuner: Web Container window, notice the Web container maximum threads is set to 50. A lower value should be specified if the HTTP server is responsible for serving a large number of static HTML and graphic files.  
  
\_\_ Set the slide bar to **Low**(value of 10).  
\_\_ Select **Next**.
- \_\_5. In the Performance Tuner: ORB Properties window, passing EJBs by reference can be selected to be used in your application. Passing by reference can greatly improve performance, however, it is important to understand how passing by reference affects your application. The setting should only be made if you are confident that changes made to passed references will not break the integrity of the application. Make no change and select **Next**.
- \_\_6. In the Performance Tuner: Data Source window, a specific datastore can be selected to be tuned in a subsequent windows.

- \_\_ From the Data Source drop down box, select **BankData**.
- \_\_ Select **Next**.
- \_\_7. Since we selected BankData as our datasource to tune, we can now set the connection pool size for the database in the Performance Tuner: Data Source window. This value should be smaller than the number of web module threads to avoid the overhead of excess database connections not being used.
- \_\_ Change the value to 50.
- \_\_ Select **Next**.
- \_\_8. In the Performance Tuner: Data Source window, the cache size for the number of prepared statements can be specified. Setting the value to the number of prepared statements in the application or the maximum number of data source connections can improve performance as statements will be accessed from the cache and not created.
- \_\_ Set the value to 1000.
- \_\_ Select **Next**.
- \_\_9. In the Performance Tuner: Databases window, you have the ability to tune the DB2 database using DB2's own Performance Tuner. The DB2 database must be version 7.2 and on a select set of operating systems. Select **Next**.
- \_\_10. In the Performance Tuner: JVM Heap Size window, the minimum and maximum sizes for the JVM heap on the application server can be specified here. It is best have your heap size match the size of your application. Too small of a heap can cause excessive garbage collection, while a large heap can take too much time to scan for stale objects.
- \_\_ Select **Next**.
- \_\_11. In the Performance Tuner: Summary window, verify the settings and select **Finish**.

- \_\_12. A warning box will be displayed indicating some settings are incorrect. The number of database connections has been set larger than the number of Web container threads and ORB threads. This can cause excess overhead as unused database connection threads are managed. The size of the prepared statement cache has also been set a level that is higher than recommend. Recommendations are based on the experiences of the WebSphere Performance Team. Select **No** to return to the wizard.

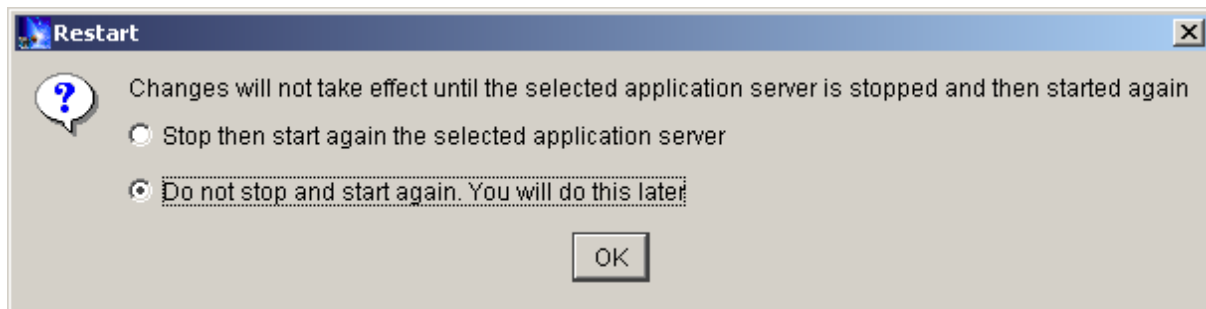


\_\_13. Use the **Back** button and set the following values in the appropriate windows:

Web container maximum threads: **30**  
Data source maximum connections: **15**  
Prepared statement cache size: **200**

\_\_ Once the values are changed, select **Finish**.

\_\_14. Almost all changes require the Admin Console as well as the Admin Server to be restarted. “Verify that do not stop and start again, You will do this later.” is selected and select **OK**.



### **What you did in this exercise**

You set the instrumentation levels for the JVM and for the EJBs within the MyBank Application using the WebSphere Advanced Admin Console. With the instrumentation levels set, you looked at the data collected using the Resource Analyzer. After make a few transactions, you were able to see a change in the values displayed in the Resource Analyzer. You next installed the Performance Servlet into the MyBankServer. You used the Performance Servlet to view the data in XML, an Open Standard. Finally, you used the Performance Tuner to change a number of settings in the application. After warned of high values by the Performance Tuner, you changed the values before setting the values through out the application server.