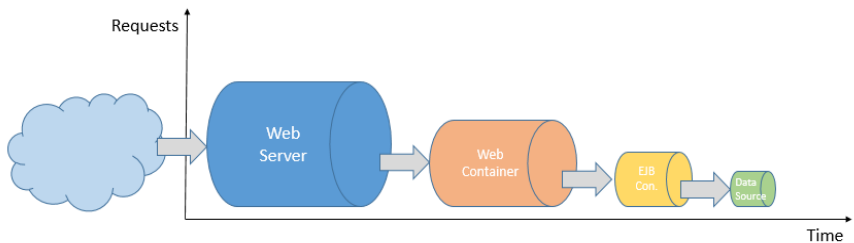


CHAPTER 15: PERFORMANCE TUNING

Theory

There are multiple areas in WebSphere Application Server to tune for better performance and there is no set of parameters that can fit into all systems. You need to consider your application, infrastructure and your workload to plan your environment. Furthermore, you need to test and analyze test results to find areas to tune.

The most common technique for performance tuning is considering queues. The idea is to keep requests as far as possible from the WebSphere to prevent overloading. To do that, it's better to keep the queues close the course of requests bigger and following queues relatively smaller.



It is not a good idea to keep queue sizes same, because we have always limited resources. Depending on your environment, you can consider having significantly bigger queues in your web server or in your data source pools. You need to analyze your architecture to understand the requirements of your environment, so that you can tune accordingly.

WebSphere Application Server uses connection pools for data source connections, because the initial connection to a database consumes higher resources. It is usually a better idea to set connection pool size lower than the “Max Connections” parameter of the “Web Container”. A deadlock occurs when application requires multiple connections per thread and the connection pool size is not large enough. In order to avoid deadlock situations, it is better to develop the application to use one connection for each thread. If this is not possible, you need to set the data source connection pool size bigger than the number of threads of which already completed their first connection and waiting for the second. Furthermore, you have to always keep in mind the number of connections that the database is configured while changing the connection pool size.

There are also tweaks to tune EJB containers for better performance. Inactive pool cleanup interval is one of the options for tuning. This setting decides the frequency of cleaning of the EJB beans from the memory. You can set this interval to smaller

values if it takes longer time to create new EJB bean. EJB cache size is another parameter to improve EJB container performance.

IBM Tivoli Performance Viewer is very helpful tool to find out optimum values for performance tuning. With this tool:

- You can view PMI data of local and remote application servers
- You can get configuration advice for better performance
- You can log the performance data for later use such as comparison of the effects of the changed parameters.
- You can view and record server performance logs.

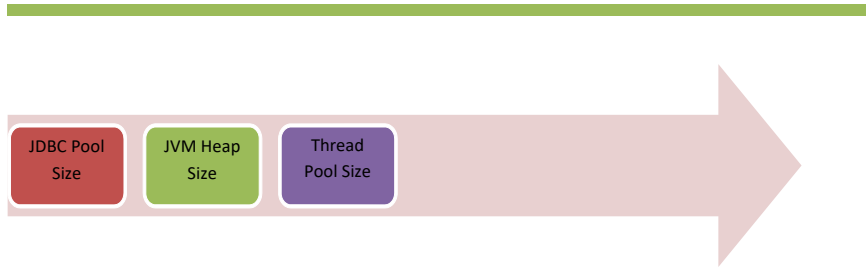
There are also other tools for analysis focused to Java such as IBM Pattern Modelling and Analysis Tool for Java Garbage Collector and IBM Monitoring and Diagnostic tools for Java.

AIM

In this lab exercise, you will be familiar with the common performance tuning parameters. In order to achieve this goal, you need to dive into following tasks:

- Change JDBC connection pool size
- Change JVM heap size
- Change web container thread pool size.

Lab Exercise 15: PERFORMANCE TUNING



1. **Change JDBC connection pool size**
2. **Change JVM heap size**
3. **Change web container thread pool size**



JDBC Pool
Size

JVM Heap
Size

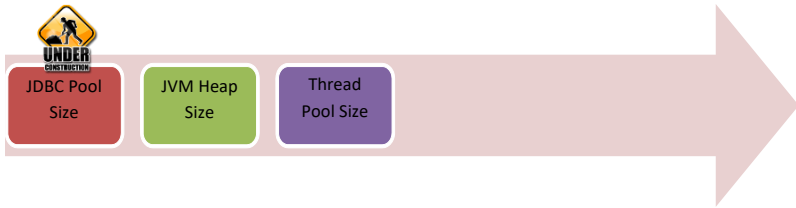
Thread
Pool Size

Task 1: Change JDBC connection pool size

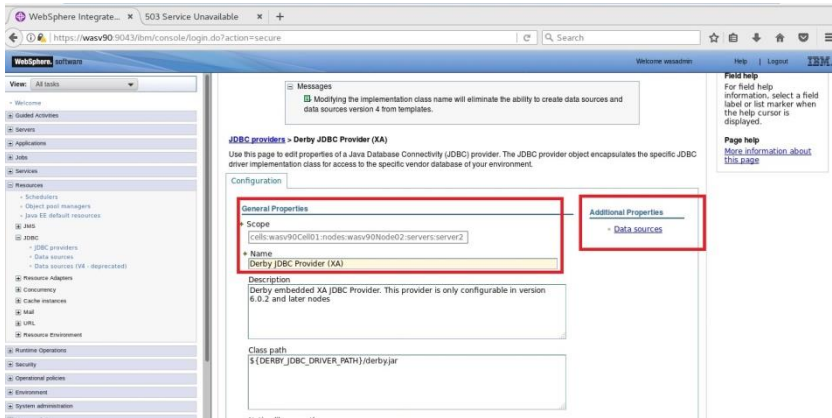
Step 1: Navigate to “Resources>JDBC>JDBC providers” and click on the name of the JDBC provider you want to configure.

WebSphere Integration Center console screenshot showing the JDBC providers configuration page. The left sidebar shows the navigation tree with 'Resources' and 'JDBC providers' highlighted. The main content area shows the 'JDBC providers' page for the scope 'Cell=wasv90Cell01, Node=wasv90Node02, Server=server2'. A dropdown menu is open, showing 'Node=wasv90Node02, Server=server2' selected. Below the dropdown, there is a table of resources. The table has columns 'Select', 'Name', 'Scope', and 'Description'. One resource is listed: ' Derby JDBC Provider (XA)' with a checkbox in the 'Select' column. The table footer shows 'Total: 1'.

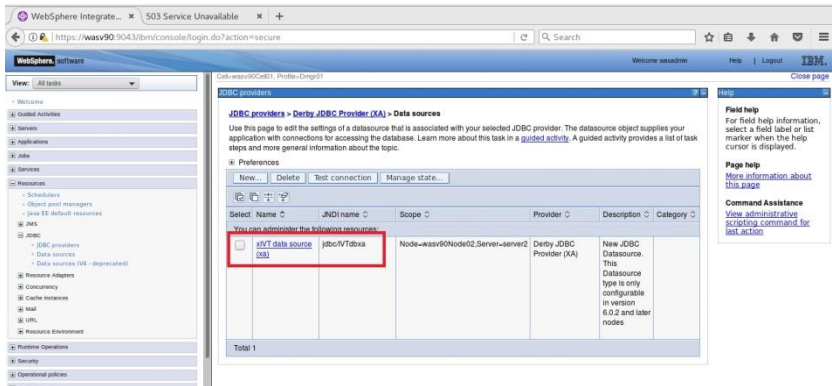
Select	Name	Scope	Description
<input type="checkbox"/>	Derby JDBC Provider (XA)	Node=wasv90Node02, Server=server2	Derby embedded XA JDBC Provider. This provider is only configurable in version 6.0.2 and later nodes



Step 2: Click on “Data sources” under “Additional Properties”.



Step 3: Click on the data source name.





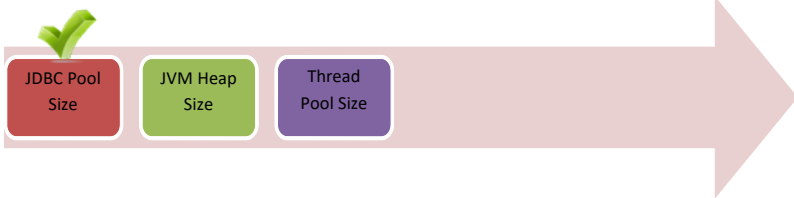
JDBC Pool
Size

JVM Heap
Size

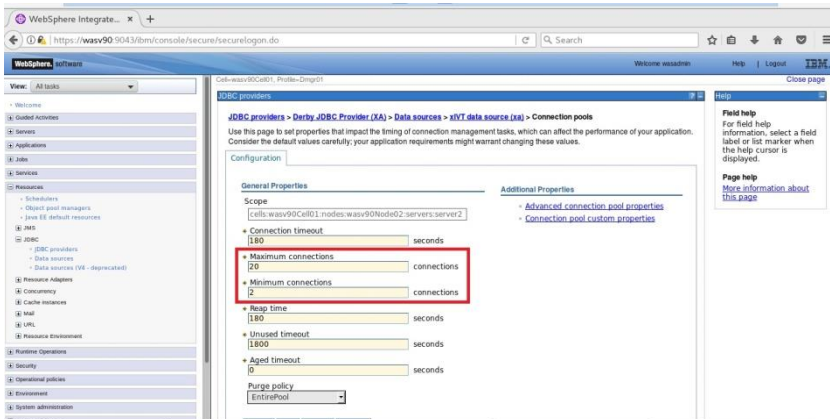
Thread
Pool Size

Step 4: Click on “Additional properties>Connection pool properties”.

The screenshot shows the WebSphere Integration Center console. The breadcrumb navigation is: **JDBC providers > Derby JDBC Provider (XA) > Data sources > JVT data source (xa)**. The main content area is titled "Configuration" and contains a "Test connection" button. Below this, the "General Properties" section is expanded, showing fields for "Scope" (cells:wasv90Cell01:modes:wasv90Node02:servers:server2), "Provider" (Derby JDBC Provider (XA)), "Name" (JVT data source (xa)), and "JNDI name" (jdbc/JVTdbxa). A checkbox "Use this data source in container managed persistence (CMP)" is checked. The "Description" field states: "New JDBC Datasource. This Datasource type is only configurable in version 6.0.2 and later nodes." On the right side, the "Additional Properties" section is expanded, showing a tree view with "Connection pool properties" selected. Other visible items in the tree are "WebSphere Application Server data source properties" and "Custom properties". The "Related Items" section on the right shows "JAAS - J2C authentication data". The left sidebar contains a navigation menu with categories like Welcome, Global Activities, Servers, Applications, Java, Services, Resources, Monitoring Operations, Security, Operational policies, Environment, and System administration.



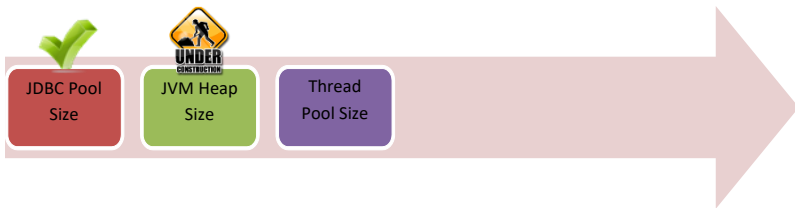
Step 5: Set the maximum and minimum number of connections, then click “OK”.



Step 6: Click “Save” to write changes to the master configuration file.



Task 1 is complete!



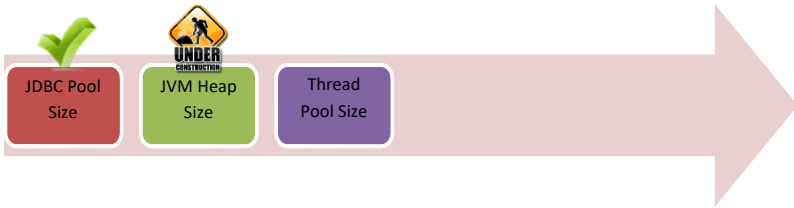
Task 2: Change JVM heap size

Step 1: Navigate to “Servers>Server Types>WebSphere application servers” and click on the name of the application server you want to configure.

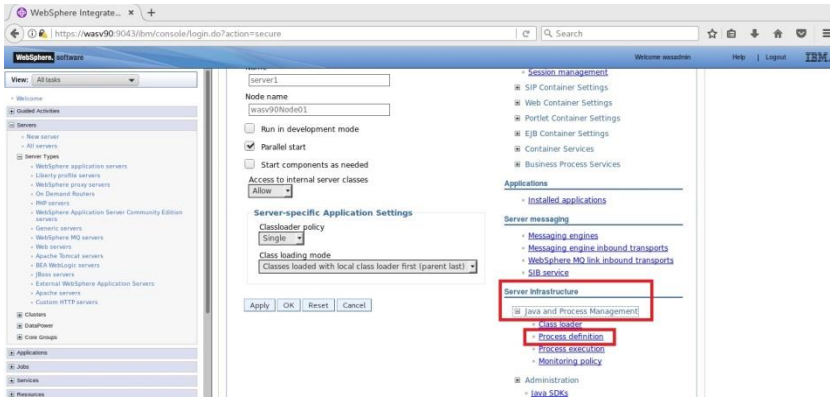
The screenshot shows the WebSphere Integrated Solutions console. The left sidebar has a navigation tree with 'Servers' and 'Server Types' highlighted. The main area displays the 'Application servers' page. The table below shows the list of application servers.

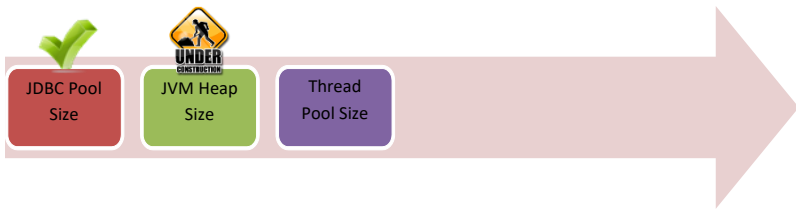
Select	Name	Node	Host Name	Version	Cluster Name	Status
<input type="checkbox"/>	App_Server01	wasv90Node01	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	App_Server02	wasv90Node02	wasv90	ND 9.0.0.0	WAS_CLUSTER	→
<input type="checkbox"/>	server1	wasv90Node01	wasv90	ND 9.0.0.0		→
<input type="checkbox"/>	server2	wasv90Node02	wasv90	ND 9.0.0.0		→

Total 4

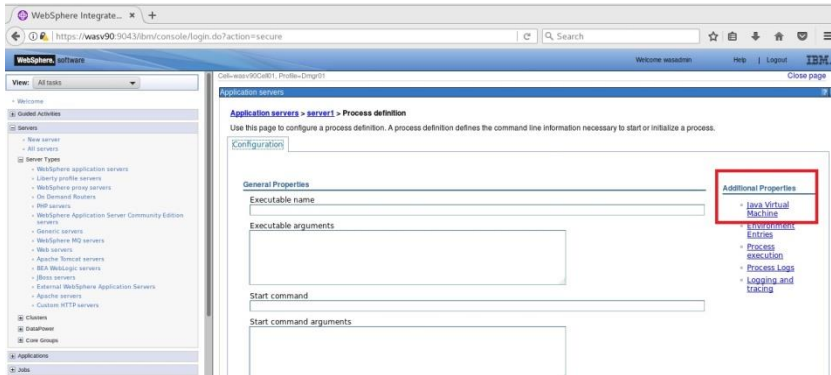


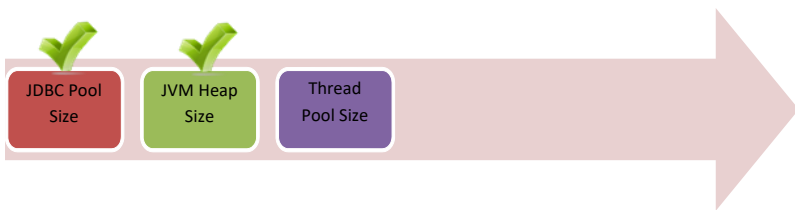
Step 2: Click on “Process definition” under “Java and Process Management”.



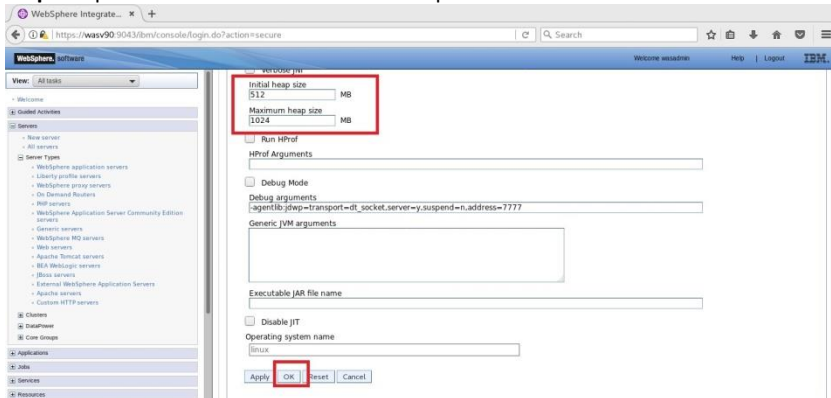


Step 3: Click “Java Virtual Machine” under “Additional Properties”.





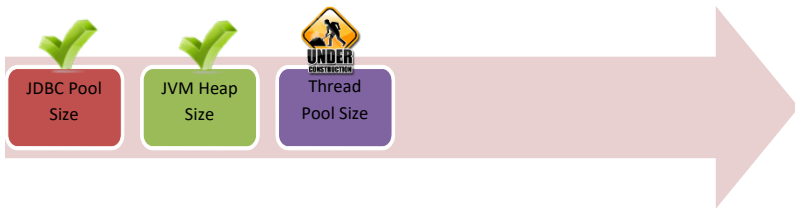
Step 4: Update the initial and maximum heap size and then click “OK”.



Step 5: Click “Save” to complete.

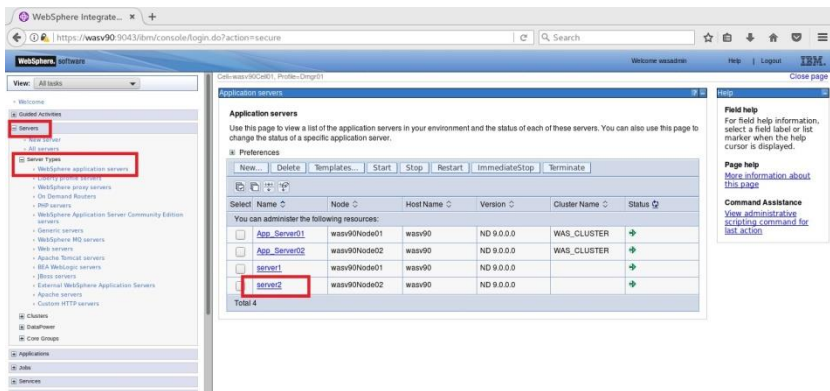


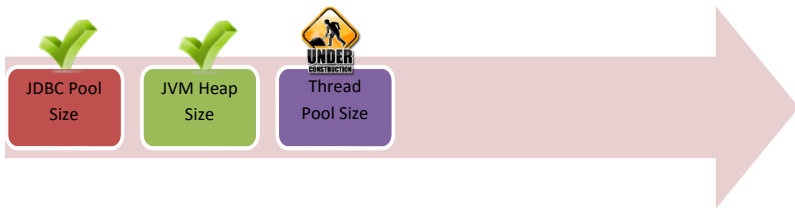
Task 2 is complete!



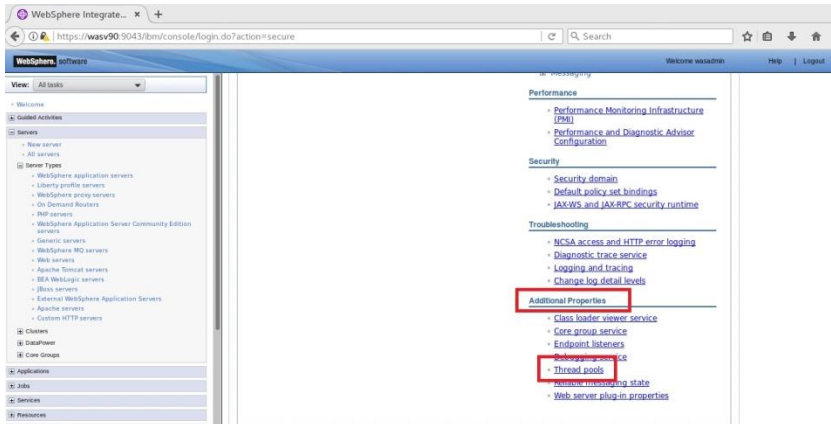
Task 3: Change web container thread pool size

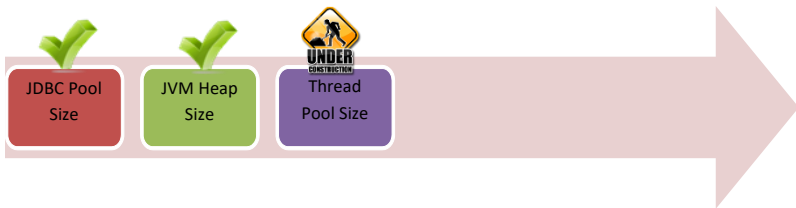
Step 1: Navigate to “Servers>Server Types>WebSphere application servers” and click on the name of the application server you want to configure.





Step 2: Click on “Thread pools” under “Additional Properties”.





Step 3: Click on “WebContainer” to configure.

The screenshot shows the WebSphere Integration console interface. The left sidebar contains a navigation tree with categories like 'Welcome', 'Global Activities', 'Servers', 'Server Types', 'Clusters', 'Applications', 'Jobs', 'Services', and 'Resources'. The main content area displays a table of thread pools. The 'WebContainer' entry is highlighted with a red rectangular box. A right-hand pane contains help information.

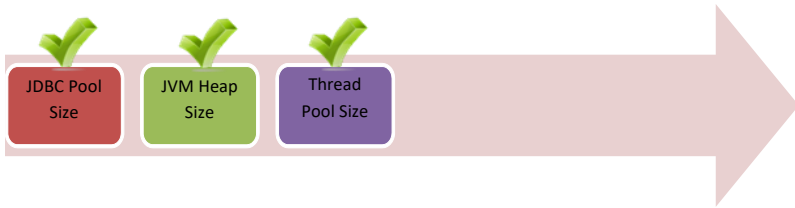
Select	Name	Description	Minimum Size	Maximum Size
<input type="checkbox"/>	Default		20	20
<input type="checkbox"/>	ORB thread pool		10	50
<input type="checkbox"/>	SBEFAPInboundThreadPool	Service Integration bus FAP inbound channel thread pool	4	50
<input type="checkbox"/>	SBEFAPThreadPool	Service Integration bus FAP outbound channel thread pool	4	50
<input type="checkbox"/>	SBJMSRAThreadPool	Service Integration Bus JMS Resource Adapter thread pool	35	41
<input type="checkbox"/>	TCPChannel DCS		20	20
<input type="checkbox"/>	WMQJCAResourceAdapter	WebSphere MQ Resource Adapter thread pool	10	50
<input checked="" type="checkbox"/>	WebContainer		50	50
<input type="checkbox"/>	server startup	This pool is used by WebSphere during server startup.	1	3

Total: 9

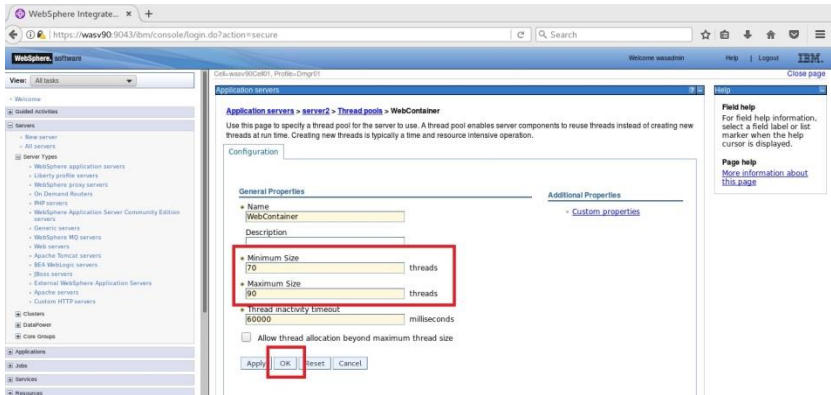
Field help
For field help information, select a field label or list marker when the help cursor is displayed.

Page help
[More information about this page](#)

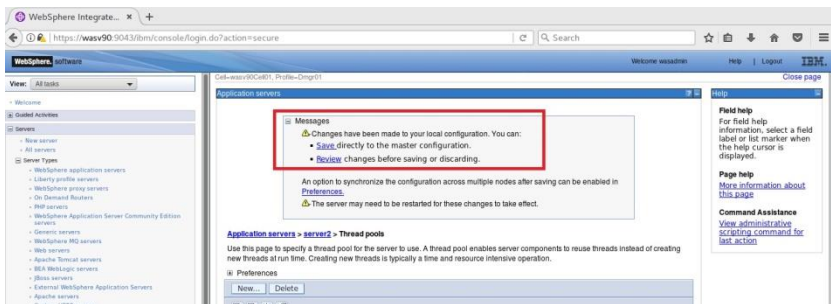
Command Assistance
[View administrative scripting command for list action](#)



Step 4: Update the number of minimum and maximum threads and click “OK”.



Step 5: Click “Save” to write changes directly to the master configuration file.



Task 3 is complete!

SUMMARY

WebSphere Application Server provides multiple areas for tuning for better performance. Better tuning comes with better knowledge of your application, infrastructure and your workload. It is a continuous cycle of testing and analyzing the test results and tuning different parts of the application server environment. For that purpose, you can use the tools that come with WebSphere Application Server or other tools which are focused to different parts. Performance tuning is one of the most critical operations since we are always limited with resources.

REFERENCES

- <http://www.ibmsoftwareservicesindia.com/basic-performance-tuning-parameters-for-websphere-application-server/>
- http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/welc_howdoi_tprf.html?lang=en

INDEX

connection pools	476
deadlock	476
EJB cache size.....	477
IBM Tivoli Performance Viewer	477
Inactive pool cleanup interval.....	477