



Setting up the application serving environment

Note

Before using this information, be sure to read the general information under “Notices” on page 373.

Compilation date: September 10, 2008

© Copyright International Business Machines Corporation 2008.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

How to send your comments	vii
Changes to serve you more quickly	ix
Chapter 1. Configuring ports	1
Chapter 2. Setting up the administrative architecture	3
Configuring cells	3
Cell settings	4
Configuring deployment managers	5
Deployment manager settings	6
Node	8
Managing nodes	9
Node collection	12
Add managed nodes	14
Node installation properties	15
Node group	16
Example: Using node groups	18
Managing node groups	19
Node group collection	20
Managing node group members	21
Node group member collection	22
Managing node agents	23
Node agent collection	24
Administrative agent	26
Administering nodes using the administrative agent	27
Administrative agent settings	29
Nodes collection for the administrative agent	30
Register or unregister with job manager	31
Job managers collection	31
Job manager	32
Administering nodes using the job manager	33
Submitting a job	34
Checking job status	38
Administering nodes of the job manager	46
Administering node resources of the job manager	48
Administering groups of nodes for the job manager	51
Configuring job managers	55
Administration service settings	57
Remote connector	57
Local connector	57
Administration services custom properties	57
com.ibm.websphere.mbeans.disableRouting	58
Administrative audits	58
Remote file services	59
Configuring remote file services	60
File transfer service settings	61
File synchronization service settings	61
Stopping or canceling the z/OS location service daemon from the MVS console	63
Determining if the z/OS location service daemon is running	64
Modifying z/OS location service daemon settings	64
z/OS location service daemon settings	65
Changing the node host names	67

Administrative topology: Resources for learning	70
Extension MBean Providers collection	70
Name	70
Description	71
Classpath	71
Extension MBean Provider settings	71
Extension MBean collection	71
Extension MBean settings	71
Java Management Extensions connector properties	72
SOAP connector and Inter-Process Communications connector properties files	79
Java Management Extensions connectors	80
Type	81
Enabled	81
JMX connector settings	81
Repository service settings	82
Audit Enabled	82
Chapter 3. Working with server configuration files	83
Configuration documents	84
Configuration document descriptions	86
Object names: What the name string cannot contain	87
Handling temporary configuration files resulting from session timeout	88
Changing the location of temporary configuration files	89
Changing the location of backed-up configuration files	89
Changing the location of the wstemp temporary workspace directory	90
Backing up and restoring administrative configuration files	91
Backing up the WebSphere Application Server for z/OS system	92
Server configuration files: Resources for learning	92
Chapter 4. Administering application servers	93
Virtual hosts	94
Configuring virtual hosts	96
Virtual host collection	98
Creating, editing, and deleting WebSphere variables	101
WebSphere variables collection	103
Introduction: Variables	104
WebSphere Variables	106
Configuring the IBM Toolbox for Java.	107
Repository service custom properties.	108
Application server custom properties for z/OS	109
Managing shared libraries	131
Creating shared libraries	132
Shared library collection	135
Associating shared libraries with applications or modules	137
Associating shared libraries with servers	139
Installed optional packages	140
Using installed optional packages	141
Library reference collection	143
Application server naming conventions	144
Test cells and production cells	144
Testing and production phases	145
Load Balancer	146
Setting up peer restart and recovery	147
Peer restart and recovery	149
Using RRS panels to resolve InDoubt units of recovery	150
Recovering with JTA XAResource managers	153

Creating application servers	154
Creating server templates	156
Deleting server templates	157
Managing application servers	157
Server collection	158
Application server settings	160
Core group service settings	171
Running servers in 31-bit mode	172
Changing the values of variables referenced in BBOM0001I messages	175
Environment entries collection	190
Starting an application server	191
Detecting and handling problems with runtime components	196
Stopping an application server	196
Automatically rejecting work requests when no servant is available to process these requests	197
Converting a 7-character server short name to 8 characters	199
Changing time zone settings	200
Web module or application server stops processing requests	217
Setting a time limit for the completion of RMI/IOP enterprise bean requests	218
Creating generic servers	220
Starting and terminating generic application servers	222
Generic server settings	223
Setting up the Server Runtime on multiple systems in a sysplex	223
Customizing base z/OS functions on the other systems in a sysplex	225
Configuring transport chains	227
Transport chains	228
HTTP transport collection	230
HTTP transport settings	230
Transport chains collection	235
Transport chain settings	235
HTTP tunnel transport channel settings	236
HTTP transport channel settings	236
TCP transport channel settings	242
DCS transport channel settings	245
ORB service transport channel settings	246
SSL inbound channel	246
Session Initiation Protocol (SIP) inbound channel settings	247
Session Initiation Protocol (SIP) container inbound channel settings	248
User Datagram Protocol (UDP) Inbound channel settings	248
Web container inbound transport channel settings	250
DataPower appliance manager transport channel settings	250
HTTP transport channel custom properties	251
HTTP Tunnel transport channel custom property	253
TCP transport channel custom properties	253
Web container transport channel custom properties	254
Configuring inbound HTTP request chunking	255
Transport chain problems	256
Deleting a transport chain	257
Disabling ports and their associated transport chains	257
Creating custom services	258
Custom service collection	260
Defining application server processes	261
Process definition settings	262
Sysplex Distributor	268
Running multiple TCP/IP stacks	268
Configuring the JVM	270
Java virtual machine settings	270

Configuring JVM sendRedirect calls to use context root	277
Java virtual machine custom properties	278
Preparing to host applications	285
Configuring multiple network interface support	286
Configuring application servers for UCS Transformation Format	288
Tuning application servers	289
Web services client to Web container optimized communication	290
Chapter 5. Balancing workloads with clusters	293
Clusters and workload management	293
Techniques for managing state	295
Workload management (WLM) for z/OS	296
Connection optimization	296
Sysplex routing of work requests	297
Address space management for work requests	298
Example of classification rules	299
Enabling multiple servants on z/OS	302
Multiple servant regions.	302
Controlling the number of servants.	303
Classifying z/OS workload	303
Sample z/OS workload classification document	307
Workload classification file.	310
Using transaction classes to classify workload for WLM	320
Creating clusters	324
Creating a cluster: Basic cluster settings	328
Creating a cluster: Create first cluster member	329
Creating a cluster: Summary settings.	331
Creating a cluster: Create additional cluster members	331
Server cluster collection	333
Enabling static routing for a cluster	335
Disabling static routing for a cluster	337
Clusters on which stateful session beans will be deployed	338
Adding members to a cluster.	339
Cluster member collection	341
Cluster member templates collection	347
Starting clusters	347
Stopping clusters	348
Replicating data across application servers in a cluster	349
Replication	350
Replication domain collection.	351
Migrating servers from multi-broker replication domains to data replication domains	353
Deleting replication domains	356
Replicating data with a multi-broker replication domain	357
Deleting clusters	362
Deleting specific cluster members	362
Configuring an application server to use the WLM even distribution of HTTP requests function	363
WLM even distribution of HTTP requests	365
WLM dynamic application environment operator commands	369
Appendix. Directory conventions	371
Notices	373
Trademarks and service marks	375

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
 1. Display the article in your Web browser and scroll to the end of the article.
 2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
 3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with `http://` work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. Configuring ports

When you configure WebSphere® Application Server resources or assign port numbers to other applications, you must avoid conflicts with other assigned ports. In addition, you must explicitly enable access to particular port numbers when you configure a firewall.

1. Review the port number settings, especially when you are planning to coexist.
2. Optional: Change the port number settings.

You can set port numbers when configuring (customizing) the product after installation. Start thinking about port numbers during the planning phase described in the "Planning for product configuration" article in the information center.

Chapter 2. Setting up the administrative architecture

You can monitor and control incorporated nodes and the resources on those nodes by using these tasks with the administrative console or other administrative tools.

About this task

After you set up the Network Deployment environment, you mainly need to monitor and control incorporated nodes and the resources on those nodes by using the administrative console or other administrative tools. Use the following tasks to perform these activities.

- Use the settings page for an administrative service to configure administrative services.
- Configure cells.
- Configure deployment managers.
- Manage nodes.
- Manage node agents.
- Manage node groups.
- Configure remote file services.
- Configure location service daemons on the z/OS® system.
- Administer job managers.
- Change the host name.
- Administer multiple application servers through an administrative agent.

Configuring cells

This topic describes how to change the cell protocol information, define custom properties for the cell, and add additional nodes.

Before you begin

Before you can configure cells, you must install the WebSphere Application Server Network Deployment product.

About this task

When you create a deployment manager profile, a cell is created. A cell provides a way to group one or more nodes of your Network Deployment product. You define the nodes that make up a cell, according to the specific criteria that make sense in your organizational environment. You probably do not need to configure the cell again.

Administrative configuration data is stored in XML files. A cell retains master configuration files for each server in every node in the cell. Each node and server also have their own local configuration files. Changes to a local node or to a server configuration file are temporary, if the server belongs to the cell. While in effect, local changes override cell configurations. Changes to the master server and master node configuration files made at the cell level replace any temporary changes made at the node when the cell configuration documents are synchronized to the nodes. Synchronization occurs at designated events, such as when a server starts.

To view information about and to manage a cell, use the settings page for a cell.

1. Access the settings page for a cell. Click **System Administration > Cell** from the navigation tree of the administrative console.

2. If the protocol that the cell uses to retrieve information from a network is not appropriate for your system, select the appropriate protocol. By default, a cell uses Transmission Control Protocol (TCP). If you want the cell to use User Datagram Protocol, select **UDP** from the list for **Cell discovery protocol** on the settings page for the cell. It is unlikely that you need to change the cell protocol configuration from TCP.
3. Click **Custom Properties** and define any name-value pairs that your deployment manager needs.
 - a. Click **New**.
 - b. Specify a name and value for the custom property.

The IBM_CLUSTER_RIPPLESTART_NOTIFICATION_TIMEOUT custom property:

Specify this custom property with an integer value in milliseconds to indicate the amount of time the ripplestart function waits for processes to shut down before restarting them. If you attempt a ripplestart and the processes have not shutdown before the start operation begins, one or more of the processes will not restart.

Property	IBM_CLUSTER_RIPPLESTART_NOTIFICATION_TIMEOUT
Data type	integer
Default	300000 milliseconds (5 minutes)

The com.ibm.websphere.management.launcher.options custom property:

Specify this custom property with a value of `displayServerInFront` to display the name of the cell, node, and server in front of the output for the `ps -ef` command. Use of this property is intended to help you identify the process ID of a server. The property has no impact on the server process.

Property	com.ibm.websphere.management.launcher.options
Data type	String
Default	None

4. When you install the WebSphere Application Server Network Deployment product, a node may have been added to the cell. You can add additional nodes on the Node page. Click **Nodes** to access the Node page, which you use to manage nodes.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but there are restrictions that apply to using both IPv4 and IPv6 in the same cell. Note that when you add a node to a cell, the format in which you specify the host name is based on the version of IP the node will be using. For further information, read the topic on IP version considerations for cells.

Results

Depending on which steps you performed, you changed the cell protocol information, defined custom properties for the cell, and added additional nodes.

What to do next

You can continue to administer your Network Deployment product by doing such tasks as managing nodes, node agents, and node groups.

Cell settings

Use this page to set the discovery protocol and address end point for an existing cell. A cell is a configuration concept, a way for an administrator to logically associate nodes according to whatever criteria make sense in the administrator's organizational environment.

To view this administrative console page, click **System Administration > Cell**.

Name

Specifies the name of the existing cell.

A cell name must be unique in any circumstance in which the product is running on the same physical machine or cluster of machines, such as a sysplex. Additionally, a cell name must be unique in any circumstance in which network connectivity between entities is required either between the cells or from a client that must communicate with each of the cells. Cell names must also be unique if their namespaces are federated. Otherwise, you might encounter symptoms such as a `javax.naming.NameNotFoundException` error, in which case, create uniquely named cells.

Short Name

Specifies the short name of the cell. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is read only. It was defined during installation and customization.

Cell Discovery Protocol

Specifies the protocol that the nodes use to contact and discover the deployment manager in the cell.

Select one of these protocol options:

UDP User Datagram Protocol (UDP)

TCP Transmission Control Protocol (TCP)

Default

TCP

Configuring deployment managers

Configure deployment managers for a single, central point of administrative control for all elements in a WebSphere Application Server distributed cell.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

Deployment managers are administrative agents that provide a centralized management view for all nodes in a cell, as well as management of clusters and workload balancing of application servers across one or several nodes in some editions. Each cell contains one deployment manager.

WebSphere Application Server for z/OS uses workload management (WLM) as the primary vehicle for workload balancing.

A deployment manager hosts the administrative console.

The Java™ virtual machine (JVM) for the deployment manager runs in 64-bit addressing mode by default. The maximum amount of virtual memory available to each JVM on the z/OS operating system in 31-bit addressing mode is 2 gigabytes (GB). However, because of product requirements, the amount of virtual memory that is available to each JVM is somewhat less. To obtain more virtual memory for the deployment

manager server, or any other server, run the server in 64-bit addressing mode. If you have applications that require large amounts of virtual memory, the applications might need to run on servers configured for 64-bit addressing mode.

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

When you create a deployment manager profile, a deployment manager is created. You can run the deployment manager with its default settings. However, you can change the deployment manager configuration settings, such as the ports that the process uses, custom services, logging and tracing settings, and so on. To view information about managing a deployment manager, use the settings page for a deployment manager.

1. Click **System Administration > Deployment manager** from the navigation tree of the administrative console to access the settings page for a deployment manager.
2. Configure the deployment manager by clicking a property, such as **Custom services**, and specifying settings.
3. If you specify the server short name as eight characters, follow the directions to convert the default seven-character short name to eight characters.
4. Optionally register or unregister the deployment manager with the job manager.
A job manager allows you to submit administrative jobs asynchronously for deployment managers and for application servers registered to administrative agents. Click **System Administration > Deployment manager**. Under Additional Properties, click **Job Managers > Register/unregister with job manager**.

Results

You configured a deployment manager with options that you selected.

What to do next

You can continue to administer your product by doing such tasks as configuring cells and managing nodes, node agents, and node groups.

Deployment manager settings

Use this page to stop the deployment manager, and to link to other pages that you can use to define additional properties for the deployment manager. A deployment manager provides a single, central point of administrative control for all of the elements in the WebSphere Application Server distributed cell.

To view this administrative console page, click **System administration > Deployment manager**.

Name

Specifies a logical name for the deployment manager. The name must be unique within the cell.

Data type String

Short name

Specifies the short name of the deployment manager server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as workload manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF) (for example, Resource Access Control Facility (RACF[®])), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeric.

The system assigns a cell-unique, default short name.

Data type String

Unique ID

Specifies the unique ID of this deployment manager server.

The unique ID property is read only. The system automatically generates the value.

Data type String

Run in 64 bit JVM mode

Specifies whether to run the deployment manager in 64-bit addressing mode to obtain more than the virtual memory that is available to the deployment manager when the deployment manager runs in 31-bit addressing mode, which is 2 gigabytes (GB).

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

Default true (checked)

Start components as needed

Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID

Specifies a string that identifies the process.

Data type String

Default None

Cell name

Specifies the name of the cell for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `Cell##` appended, where `##` is a two-digit number.

Data type String

Default `host_nameCell01`

Node name

Specifies the name of the node for the deployment manager. The default is the name of the host computer on which the deployment manager is installed with `CellManager##` appended, where `##` is a two-digit number.

Data type String

Default `host_nameCellManager01`

State

Indicates the state of the deployment manager. The state is *Started* when the deployment manager is running and *Stopped* when the deployment manager is not running.

Data type	String
Default	Started

Node

A *node* is a logical grouping of managed servers.

A node usually corresponds to a logical or physical computer system with a distinct IP host address. Nodes cannot span multiple computers.

By default, node names are based on the host name of the computer, for example MyHostName01.

Nodes can be managed or unmanaged. An unmanaged node does not have a node agent or administrative agent to manage its servers, whereas a managed node does. Both application servers and supported Web servers can be on unmanaged or managed nodes.

A stand-alone application server is an unmanaged node. The application server node becomes a managed node when it is either federated into a cell or registered with an administrative agent.

When you create a managed node by federating the application server node into a deployment manager cell, a node agent is automatically created. The node agent process manages the application server configurations and servers on the node.

When you create a managed node by registering an application server node with an administrative agent, the application server must be an unfederated application server node. The administrative agent is a single interface that monitors and controls one or more application server nodes so that you can use the application servers only to run your applications. Using a single interface reduces the overhead of running administrative services in every application server.

A managed node in a cell can have WebSphere Application Servers, Java Message Service (JMS) servers (on Version 5 nodes only), Web servers, or generic servers. A managed node that is not in a cell, but is instead registered to an administrative agent, can have application servers, web servers, and generic servers on the node.

An unmanaged node can exist in a cell as long as the unmanaged node only has a supported Web server defined on it. Unsupported Web servers can be on unmanaged nodes only and cannot be in a cell.

You can use the command line only to create a managed node that is registered to an administrative agent.

You can create a managed node in a cell in one of the following ways:

- Administrative console
- Command line
- Administrative script
- Java program

Each of these methods for adding a node to a Network Deployment cell includes the option of specifying a target node group for the managed node to join. If you do not specify a node group, or you do not have the option of specifying a node group, the default node group of DefaultNodeGroup is the target node group.

On the z/OS system, the default DefaultNodeGroup node group is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. A z/OS system node from a different sysplex cannot be a member of this node group and must be a member of a sysplex node group for its sysplex.

Whether you specify an explicit node group for a cell or accept the default, the node group membership rules must be satisfied. If the node that you are adding does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.

Each managed node that is joined to a cell must be a member of a node group. However, a managed node that is registered to an administrative agent cannot be a member of a node group.

The concepts of managed and unmanaged nodes are not applied to the registration of nodes to the job manager.

Administrative functions for Web server nodes

WebSphere Application Server supports basic administrative functions for all supported Web servers. For example, the generation of a plug-in configuration can be performed for all Web servers. However, propagation of a plug-in configuration to remote Web servers is supported only for IBM® HTTP Servers that are defined on an unmanaged node. If the Web server is defined on a managed node, propagation of the plug-in configuration is done for all the Web servers by using node synchronization. The Web server plug-in configuration file is created according to the Web server definition and is based on the list of applications that are deployed on the Web server. You can also map all supported Web servers as potential targets for the modules during application deployment.

WebSphere Application Server supports some additional administrative console tasks for IBM HTTP Servers on managed and unmanaged nodes. For instance, you can start IBM HTTP Servers, stop them, terminate them, display their log files, and edit their configuration files.

Managing nodes

This topic describes how to add a node, select the discovery protocol for a node, define a custom property for a node, stop servers on a node, and remove a node.

Before you begin

A node is a grouping of managed or unmanaged servers. You can add both managed and unmanaged nodes to the WebSphere Application Server topology. If you add a new node for an existing WebSphere Application Server to the Network Deployment cell, you add a managed node. If you create a new node in the topology for managing Web servers or servers other than WebSphere Application Servers, you add an unmanaged node.

To view information about nodes and managed nodes, use the Nodes page. To access the Nodes page, click **System Administration > Nodes** in the administrative console navigation tree.

About this task

You can manage nodes on an application server through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.

- **Add a node.**

1. Go to the Nodes page and click **Add Node**. Choose whether you want to add a managed or unmanaged node, and click **Next**.

2. For a managed node, verify that an application server is running on the remote host for the node that you are adding. On the Add Node page, specify a host name, connector type, and port for the application server at the node you are adding.
3. For a managed node, perform one of the following sets of actions listed in the table:

If the deployment manager is on	And the node that you add to the cell is on	Complete the appropriate set of actions:
A z/OS system	A z/OS system and is in the same sysplex as the deployment manager	Optionally specify a node group and a core group. Click OK .
A z/OS system	A z/OS system, but is on a different sysplex than the deployment manager	Specify a node group that contains nodes from the same sysplex as the node you are adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click OK .
The distributed platform or the i5/OS® platform	A z/OS system	Specify a node group that contains nodes from the same sysplex as the node you are now adding. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click OK .
A z/OS system	The distributed platform or the i5/OS platform	Specify a node group that contains distributed nodes. If no such node group exists, create a node group and then specify that node group. Optionally specify a core group. Click OK .

For the node group option to display, a group other than the default node group must first be created. Likewise, for the core group option to display, a group other than the default core group must first be created.

4. For managed nodes, another administrative console panel is displayed if the node to federate is on a Windows® operating system. Specify on the panel whether you want to register the node agent to run as a Windows service. If security is enabled, you can optionally enter the local operating system user name and password under which you will run the service. If you do not specify a user name and password, the service runs under the local system identity. When you run remove the node, the node agent is de-registered as a Window service.
5. For an unmanaged node, on the **Nodes > New** page, specify a node name, a host name, and a platform for the new node. Click **OK**.

The node is added to the WebSphere Application Server environment and the name of the node is displayed in the collection on the Nodes page.

Join subsequent WebSphere Application Server for z/OS nodes from the same sysplex to the same sysplex node group. If you add WebSphere Application Server for z/OS nodes from different sysplexes to the same cell, establish a separate sysplex node group for the nodes of each sysplex. On completing this step, you will have added one or more nodes.

Note: When nodes are added while LDAP security is enabled, the following exception is generated in the deployment manager System.out log under certain circumstances. If this happens, restart the deployment manager to resolve the problem.

```
0000004d ORBRas E com.ibm.ws.security.orbssl.WSSSLClientSocketFactoryImpl
createSSLsocket ProcessDiscovery : 0 JSSL0080E: javax.net.ssl.SSLHandshakeException -
The client and server could not negotiate the desired level of security.
Reason?com.ibm.jsse2.util.h: No trusted certificate found
```

- **Select the discovery protocol.**

If the discovery protocol that a node uses is not appropriate for the node, select the appropriate protocol. On the Nodes page, click the node to access the Settings for the node. Select a value for **Discovery protocol**. User Datagram Protocol (UDP) is faster than Transmission Control Protocol (TCP). However, TCP is more reliable than UDP because UDP does not guarantee the delivery of datagrams to the destination. The default of TCP is the recommended value.

For a node agent or deployment manager, use **TCP** or **UDP**.

A managed process uses multicast as its discovery protocol. The discovery protocol is fixed for a managed process. The main benefit of using multicast on managed processes is efficiency for the node agent. Suppose you have forty servers in a node. A node agent that uses multicast sends one broadcast to all forty servers. If a node agent did not use multicast, it would send discovery queries to all managed processes one at a time, totaling forty sends. Additional benefits of using multicast are that you do not have to configure the discovery port for each server or prevent port conflicts because all servers in one node listen to one port instead of to one port for each server.

- **Define a custom property for a node.**

1. On the Nodes page, click the node for which you want to define a custom property.
2. On the Settings for the node, click **Custom Properties**.
3. On the Property collection page, click **New**.
4. On the Settings page for a property instance, specify a name-value pair and a description for the property, and click **OK**.

- Synchronize the node configuration.

If you add a managed node or change a managed node configuration, synchronize the node configuration. On the Node Agents page, ensure that the node agent for the node is running. Then, on the Nodes page, select the check box beside the node whose configuration files you want to synchronize and click **Synchronize** or **Full Resynchronize**.

Clicking either option sends a request to the node agent for that node to perform a configuration synchronization immediately, instead of waiting for the periodic synchronization to occur. This action is important if automatic configuration synchronization is disabled, or if the synchronization interval is set to a long time, and a configuration change is made to the cell repository that needs to replicate to that node. Settings for automatic synchronization are on the File Synchronization Service page.

Synchronize requests that a node synchronization operation be performed using the normal synchronization optimization algorithm. This operation is fast, but might not fix problems from manual file edits that occur on the node. It is still possible for the node and cell configuration to be out of synchronization after this operation is performed.

Full Resynchronize clears all synchronization optimization settings and performs configuration synchronization anew, so there is no mismatch between node and cell configuration after this operation is performed. This operation can take longer than the **Synchronize** operation.

Unmanaged nodes cannot be synchronized.

- **Stop servers on a node.**

On the Nodes page, Select the check box beside the managed node whose servers that you want to stop running, and click **Stop**.

- **Remove a node.**

On the Nodes page, Select the check box beside the node that you want to delete and click **Remove Node**. If you cannot remove the node by clicking **Remove Node**, remove the node from the configuration by clicking **Force Delete**.

- **View node capabilities.**

Review the node capabilities, such as the product version through the administrative console. You can also query them through the Application Server application programming interface (API) or the wsadmin tool. For information on the wsadmin tool, see the *Using the administrative clients* PDF.

The product versions for WebSphere Application Server are as follows: The base edition of WebSphere Application Server is listed in the version column as Base. The express edition of WebSphere Application Server is listed in the version column as Express. The Network Deployment product is listed in the version column as ND.

Node collection

Use this page to manage nodes in the WebSphere Application Server environment. Nodes group managed servers. The table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to the list by clicking **Add Node**.

To view this administrative console page, click **System administration > Nodes**.

Name

Specifies a name for a node that is unique within the cell.

A node corresponds to a physical computer system with a distinct IP host address. The node name is usually the same as the host name for the computer.

Version

Specifies the product name and version number of the node.

The product version is the version of a WebSphere Application Server for managed nodes.

For unmanaged nodes on which you can define Web servers, the version displays as not applicable

The base edition of WebSphere Application Server is listed in the version column as Base. The express edition of WebSphere Application Server is listed in the version column as Express. The Network Deployment product is listed in the version column as ND.

The product in the version column indicates the product that you used to create the profile, not the type of profile that you installed. For example, if you use the Network Deployment product to install a profile type of application server, the version column indicates ND.

Discovery protocol

Specifies the protocol that servers use to discover the presence of other servers on this node.




The possible protocol options follow:

UDP User Datagram Protocol (UDP)

TCP Transmission Control Protocol (TCP)

Status

Indicates that the node is either synchronized, not synchronized, unknown, or not applicable.

	Synchronized	The configuration files on this node are synchronized with the deployment manager.
	Not synchronized	The configuration files on this node are not synchronized with the deployment manager and are out-of-date. Perform a synchronize operation to get the latest configuration changes on the node.
	Unknown	The state of the configuration file cannot be determined because the node agent cannot be reached for this node.
	Not applicable	The status column is not applicable for this node because the node is an unmanaged node.

Node settings

Use this page to view or change the configuration or topology settings for either a managed node instance or an unmanaged node instance.

A managed node is a node with an Application Server and a node agent that belongs to a cell. An unmanaged node is a node defined in the cell topology that does not have a node agent running to manage the process. Unmanaged nodes are typically used to manage Web servers.

To view this administrative console page, click **System administration > Nodes > node_name**.

Name:

Specifies a logical name for the node. The name must be unique within the cell.

A node name usually is identical to the host name for the computer. However, you choose the node name. You can make the node name some name other than the host name.

Data type String

Short Name:

Specifies the name of a node. The name is 1-8 characters, alphanumeric or national language. It cannot start with a numeric.

The short name property is defined during installation and customization. However, you can change the short name using the **renameNode.sh** command.

Host name:

Specifies the host name of the unmanaged node that is added to the configuration.

Date type String
Default None

Discovery Protocol:

Specifies the protocol that the node follows to retrieve information from a network. The Discovery protocol setting is only valid for managed nodes.

Select from one of these protocol options:

UDP User Datagram Protocol (UDP)
TCP Transmission Control Protocol (TCP)

Data type String
Default TCP
Range Valid values are UDP or, TCP.

UDP is faster than TCP, but TCP is more reliable than UDP because UDP does not guarantee delivery of datagrams to the destination. Between these two protocols, the default of TCP is recommended.

File permissions: Specifies the most lenient file permissions for the application files that WebSphere Application Server extracts into the application destination location. A deployer can override the permissions by configuring the permissions at the application level. However, if the file permissions specified at the application level are more lenient than the ones specified at the node, the ones specified

at the node are used. The File permissions setting is only valid for managed nodes.

Data type	String
Default	755 , or rwx-rx-rx, for files that end in .dll, .so, .a and .sl if no value is set

Platform type:

Specifies the operating system on which the unmanaged node runs.

Valid options are:

Windows

AIX®

HP-UX

Solaris

Linux®

OS/400®

z/OS

Add managed nodes

A managed node is a node with an application server and a node agent that belongs to a deployment manager cell. Use this page to add an application server node to a deployment manager cell.

To view this deployment manager administrative console page, click **System Administration > Nodes > Add node > Next** .

Node connection

Specifies connection information for WebSphere Application Server.

- **Host**

Specifies the host name or IP address of the node to add to the cell. A WebSphere Application Server instance must be running on this machine.

Data type	String
Default	None

- **JMX connector type**

Specifies the Java Management Extensions (JMX) connectors that communicate with the WebSphere Application Server when you invoke a scripting process.

Select from one of these JMX connector types:

Simple Object Access Protocol (SOAP)

Use when the Application Server connects to a SOAP server.

Remote Method Invocation (RMI)

Use when the Application Server connects to an RMI server.

- **JMX connector port**

Specifies the port number of the JMX connector on the instance to add to the cell. The default SOAP connector port is 8880.

Date type	Integer
Default	8880

- **Application server user name**

Specifies the administration user name that connects to the remote Application Server whose node is being added to the cell. The Application Server user name and password are used to connect to the Application Server and start the add node process at the Application Server. The Application Server user name and password settings always display. You must specify values for them if security is enabled at the Application Server. Otherwise, leave them blank.

- **Application server password**

Specifies the password for the Application Server user name that you supply.

- **Deployment manager user name**

Specifies the deployment manager administration user name that the Application Server uses when connecting to the deployment manager to add its node to the cell. The deployment manager user name and password settings display only if security is enabled at the deployment manager. The deployment manager user name and password are required if their settings display.

- **Deployment manager password**

Specifies the password for the deployment manager user name that you supply.

Options

Select from the following settings to further specify characteristics when adding a managed node to a cell.

- **Include Applications**

Copies the applications installed on the remote instance into a cell. If the applications to copy have the same name as the applications that currently exist in the cell, the Application Server does not copy the applications.

- **Include buses**

Specifies whether to move the bus configuration at the node to the deployment manager.

- **Starting port**

Specifies the port numbers for the node agent process.

Use default

Specifies whether to use the default node agent port numbers.

Specify

Allows you to specify the starting port number in the Port number field. WebSphere Application Server administration assigns the port numbers in order from the starting port number. For example, if you specify 9950, the administration program configures the node agent ports as 9950, 9951, 9952, and so on.

- **Core Group**

Specifies the group to which you can add a cluster or node agent. By default, clusters or node agents are added to the DefaultCoreGroup group.

Select from one of the core groups if a list is displayed. The list displays if a core group in addition to the default core group exists.

- **Node group**

Specifies the group to which you can add the node. By default, nodes are added to the DefaultNodeGroup group.

Select from one of the node groups if a list is displayed. The list displays if a node group in addition to the default node group exists.

Node installation properties

Use this page to view read-only installation properties for this node. These properties provide information about the capabilities of the node that are collected during product installation time, such as the operating system name, architecture and version, or WebSphere Application Server product levels that are installed on the node.

To view this administrative console page, click **System administration > Nodes > *node name* > Node installation properties**.

Information about a node, such as operating system platform and product features, is maintained in the configuration repository in the form of properties. As product features are installed on a node, new property settings are added.

WebSphere Application Server system management uses the managed object metadata properties as follows:

- To display the node version in the administrative console
- To ensure that new configuration types or attributes are not created or set on older release nodes
- To ensure that new resource types are not created on old release nodes
- To ensure that new applications are not installed on old release nodes because the old run time cannot support the new applications

For detailed information about the following properties, see the Application Server application programming interface (API).

com.ibm.websphere.baseProductShortName

The product short name for the WebSphere Application Server that is installed.

com.ibm.websphere.baseProductVersion

The version of WebSphere Application Server that is installed.

com.ibm.websphere.nodeOperatingSystem

The operating system platform on which the node runs.

com.ibm.websphere.nodeSysplexName

The sysplex name on a z/OS operating system.

Node group

A *node group* is a collection of managed nodes. Managed nodes are WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

- “Node groups”
- “Sysplex node groups” on page 17
- “Example: Using node groups” on page 17

Node groups

Nodes that you organize into a node group need to be similar in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere Application Server administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default `DefaultNodeGroup` node group.

A node can be a member of more than one node group.

Nodes on distributed platforms and the i5/OS platform cannot be members of a node group that contains a node on a z/OS platform. However, nodes on distributed platforms and nodes on the i5/OS platform can be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

An Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can be in one sysplex node group only.

When the deployment manager is configured on a z/OS node, the default node group, `DefaultNodeGroup`, is the sysplex node group for the deployment manager node and any other node in the cell from the same sysplex. Sysplex node groups are special node groups that the system manages.

Sysplex node groups

A sysplex node group is a node group unique to the z/OS operating system. The sysplex node group includes a sysplex name and a z/OS operating system location service configuration. A sysplex is a collection of z/OS systems that cooperate by using certain hardware and software products to process workloads.

You cannot explicitly create a sysplex node group. The z/OS operating system creates sysplex node groups in the following ways:

- When you configure a deployment manager server on the z/OS operating system, the default node group is a sysplex node group. The deployment manager is automatically a member of the sysplex node group. Application Server for z/OS nodes that you add to the network deployment cell are automatically members of this node group.
- You can add an Application Server for z/OS node to a network deployment cell whose deployment manager is on a distributed platform node. In this case, you must add the first Application Server for z/OS node for the network deployment cell to an empty node group. The system automatically configures the node group into a sysplex node group by using the sysplex name and the z/OS location service configuration that belongs to the Application Server for z/OS node.

You cannot remove a node from a sysplex node group. However, if a node is the only member of a sysplex node group, you can add that node to an empty node group. The empty node group is converted into a sysplex node group and the former sysplex node group of the node is converted into a regular node group.

You cannot delete a node group that is a sysplex node group.

Example: Using node groups

By organizing nodes that satisfy your application requirements into a node group, you establish an administrative policy that governs which nodes can be used together to form a cluster. The people who define the cell configuration and the people who create server clusters can operate with greater independence from one another, if they are different people.

Example 1

Assume the following information:

- A cell is comprised of nodes one to eight.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes six, seven, and eight are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or some combination of distributed platform nodes and i5/OS platform nodes.
- By default, all the nodes are in the default `DefaultNodeGroup` node group.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes six, seven, and eight. Therefore, clusters that host these applications can be formed only on nodes six, seven, and eight. To define a clustering policy that guides users of your WebSphere cell into building clusters that can span only predetermined nodes, create an additional node group called `WBINodeGroup`, for example. Add to the node group nodes six, seven, and eight. If you create a cluster on a node from the `WBINodeGroup` node group, the system allows only nodes from the `WBINodeGroup` node group to be members of the cluster.

Example 2

Assume the following information:

- A cell is comprised of nodes one to six.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes one to four are some combination of distributed platform nodes and i5/OS platform nodes.
- Nodes five and six are nodes on the z/OS operating system and are in the `PLEX1` sysplex.
- The deployment manager is on a distributed platform node.
- Nodes one to four are members of the `DefaultNodeGroup` node group by default.
- You created empty `PLEX1NodeGroup` node group to group the z/OS operating system nodes on the `PLEX1` sysplex.
- You joined the nodes on the z/OS operating system to the `PLEX1NodeGroup` node group when you added them to the cell. Nodes on the z/OS operating system cannot be in the same node group with the distributed platform nodes.

Applications that exploit z/OS functions in the `PLEX1` sysplex can run successfully on nodes five and six only. Therefore, clusters that host these applications can be formed only on nodes five and six. The required separation of distributed platform nodes and i5/OS platform nodes from z/OS system nodes establishes a natural clustering policy that guides users of your Application Server cell into building clusters that can span only predetermined nodes. If you create a cluster on a node from the `PLEX1NodeGroup` node group, the system allows only nodes from the `PLEX1NodeGroup` node group to be members of the cluster.

Example: Using node groups

Use node groups to define groups of nodes that are capable of hosting members of the same cluster. An application that is deployed to a cluster must be capable of running on any of the cluster members. The node that hosts each of the cluster members must be configured with software and settings that are necessary to support the application.

By organizing nodes that satisfy your application requirements into a node group, you establish an administrative policy that governs which nodes can be used together to form a cluster. The people who define the cell configuration and the people who create server clusters can operate with greater independence from one another, if they are different people.

Example 1

Assume the following information:

- A cell is comprised of nodes one to eight.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes six, seven, and eight are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or some combination of distributed platform nodes and i5/OS platform nodes.
- By default, all the nodes are in the default `DefaultNodeGroup` node group.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes six, seven, and eight. Therefore, clusters that host these applications can be formed only on nodes six, seven, and eight. To define a clustering policy that guides users of your WebSphere cell into building clusters that can span only predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes six, seven, and eight. If you create a cluster on a node from the WBINodeGroup node group, the system allows only nodes from the WBINodeGroup node group to be members of the cluster.

Example 2

Assume the following information:

- A cell is comprised of nodes one to six.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes one to four are some combination of distributed platform nodes and i5/OS platform nodes.
- Nodes five and six are nodes on the z/OS operating system and are in the PLEX1 sysplex.
- The deployment manager is on a distributed platform node.
- Nodes one to four are members of the DefaultNodeGroup node group by default.
- You created empty PLEX1NodeGroup node group to group the z/OS operating system nodes on the PLEX1 sysplex.
- You joined the nodes on the z/OS operating system to the PLEX1NodeGroup node group when you added them to the cell. Nodes on the z/OS operating system cannot be in the same node group with the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully on nodes five and six only. Therefore, clusters that host these applications can be formed only on nodes five and six. The required separation of distributed platform nodes and i5/OS platform nodes from z/OS system nodes establishes a natural clustering policy that guides users of your Application Server cell into building clusters that can span only predetermined nodes. If you create a cluster on a node from the PLEX1NodeGroup node group, the system allows only nodes from the PLEX1NodeGroup node group to be members of the cluster.

Managing node groups

This task discusses how to create and manage node groups.

Before you begin

Read about Nodes groups if you are unfamiliar with them.

About this task

Your WebSphere Application Server environment has a default node group. However, if you need additional node groups to manage your Application Server environment, you can create and configure additional node groups. You can delete a node group as long as it is not a default node group.

- View and configure node groups.
 1. Click **System Administration > Node groups** in the console navigation tree.
 2. To view additional information about a particular node group or to further configure a node group, click on the node group name under **Name**.
- Create a node group.
 1. Click **System Administration > Node groups** in the console navigation tree.
 2. Click **New**.
 3. Specify the node group name and description.

The node group is added to the WebSphere Application Server environment . The name of the node group appears in the name column of the Node group page.

You can now add nodes to the node group. See “Managing nodes” on page 9 and “Managing node group members” on page 21 for information on how to add the nodes.

- Delete a node group if the node group is not the default node group.
 1. If the node group contains members, delete the members:
 - a. Click **System Administration > Node groups** in the console navigation tree.
 - b. Under **Name**, click the node group whose members you want to delete.
 - c. Click **Node group members**.
 - d. Select all the node group members.
 - e. Click **Remove**.
 2. Click **System Administration > Node groups**.
 3. Select an empty node group.
 4. Click delete.

Node group collection

Use this page to manage node groups. A node group is a collection of WebSphere Application Server nodes. A node group defines a boundary for server cluster formation.

Nodes that are organized into a node group should be enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster. The deployment manager does no validation to guarantee that nodes in a given node group have anything in common.

Node groups are optional and are established at the discretion of the WebSphere administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default node group. The default node group is DefaultNodeGroup.

A node can be a member of more than one node group.

An Application Server node must be a member of a sysplex node group. Nodes in the same sysplex must be in the same sysplex node group. A node can only be in one sysplex node group. Sysplex node groups are special node groups that the system manages.

Nodes on distributed platforms and the i5/OS platform cannot be members of a node group that contains a node on a z/OS platform. However, nodes on distributed platforms and nodes on the i5/OS platform can be members of the same node group.

To delete a node group, the node group must be empty. The default node group cannot be deleted.

To view this administrative console page, click **System Administration > Node groups**.

Name

Specifies a name for a node group that is unique within the cell.

Members

Specifies the number of members or nodes in the node group.

Description

Specifies a description that you define for the node group.

Node group settings

Use this page to view or change the configuration or topology settings for a node group instance.

To view this administrative console page, click **System Administration > Node groups > node group name**.

Name:

Specifies a logical name for the node group. The name must be unique within the cell. The name can start with a number.

Data type	String
Maximum length	64 characters

Short name:

Specifies the name of a node. The name must contain 1-8 characters, which are either alphanumeric or national language. It cannot start with a number.

On the z/OS system the short name property is:

- Read-only
- Used only by sysplex node groups
- Defined during installation and customization

Sysplex:

Specifies the name of a node. The name is eight characters, alphanumeric or national language. It cannot start with a numeric. It is used only by sysplex node groups on the z/OS platform. It is defined during installation and customization on z/OS platforms only.

The Sysplex property is read only.

Members:

Specifies the number of nodes within the node group.

Data type	Integer
------------------	---------

Description:

Specifies the description that you define for the node group. The description has no specific maximum length.

Managing node group members

Use this topic to manage the nodes in your node groups by viewing, adding or deleting the nodes in a node group.

Before you begin

Read about Nodes groups if you are unfamiliar with them.

About this task

Make the nodes that you organize into a node group enough alike in terms of installed software, available resources, and configuration to enable servers on those nodes to host the same applications as part of a server cluster.

Node group membership must adhere to the following rules:

- A node in a node group must be a managed node.
- A managed node must be a member of at least one node group. Initially, all WebSphere Application Server nodes are members of the default node group named DefaultNodeGroup.
- If the node is on the z/OS platform, the node must be a member of a sysplex node group. The node can also be a member of other node groups that are not sysplex node groups.
- Nodes on distributed platforms and i5/OS platforms cannot be members of a node group that contains a node on a z/OS platform.
- Nodes that are in different sysplexes must be members of different node groups. Nodes that are in the same sysplex must belong to the same node group.
- View node groups members.
 1. Click **System Administration > Node groups > *node group name* > Nodes > Node group members** in the console navigation tree.
 2. To view additional information about a particular node group member for this node group, click on the node group member name under **Name**.
- Add a node to a node group.
 1. Click **System Administration > Node groups > *node group name* > Nodes > Node group members** in the console navigation tree.
 2. Click **Add**.
 3. Select the node from a list. The node group member name is the node name.

The node group member is added to the node group specified on the breadcrumb trail. The name of the node group member appears in the name column of the Node group member page. You can add additional nodes of similar characteristics to the node group by repeating the steps for adding a node to a node group.

If the node you add does not satisfy the node group membership rules for the target node group, the add node operation fails with an error message.

- Remove a node from a node group.
 1. Click **System Administration > Node groups > *node group name* > Nodes > Node group members** in the console navigation tree.
 2. Select the box next to each node group member that you want to remove from the node group.
 3. Click **Remove**.

Each node group member that you selected is removed from the node group specified on the breadcrumb trail.

Node group member collection

Use this page to manage node groups members. A node group member is a WebSphere Application Server node.

Click **Add** to add node members to the node group. Click **Remove** to remove node members from the node group.

To view this administrative console page, click **System Administration > Node groups > *node group name* > Node group members**.

Name

Specifies the name of a node group member.

Node group member settings

Use this page to view or change the configuration or topology settings for a node group member.

To view this administrative console page, click **System Administration > Node groups > node group name > Node group members > node group member name**.

Name:

Specifies a logical name for the node group member. A node group member is a node. The name must be unique within the cell.

A node group member name usually is identical to the host name for the computer.

Data type	String
Maximum length	64 characters

The name must contain alphanumeric or national language characters and can start with a number.

Managing node agents

Node agents are administrative agents that represent a node to your system and manage the servers on that node. Node agents monitor application servers on a host system and route administrative requests to servers.

Before you begin

Before you can manage a node agent, you must install the Network Deployment product.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

A node agent is a server that is created automatically when a node is added to a cell. A node agent runs on every host computer system that participates in the Network Deployment product. You can view information about a node agent, stop and start the processing of a node agent, stop and restart application servers on the node that is managed by the node agent, and so on.

A node agent is purely an administrative agent and is not involved in application serving functions. A node agent also hosts other important administrative functions, such as file transfer services, configuration synchronization, and performance monitoring.

You can manage nodes through the wsadmin scripting tool, through the Java application programming interfaces (APIs), or through the administrative console. Perform the following tasks to manage nodes on an application server through the administrative console.

- View information about a node agent. Click **System Administration > Node agents** in the console navigation tree. To view additional information about a particular node agent or to further configure a node agent, click the node agent name under **Name**.
- Stop and then restart all of the application servers on the node that is managed by the node agent. On the Node Agents page, select the check box beside the node agent that manages the node with servers that you want to restart, and click **Restart all servers on node**.

Clicking **Restart all servers on node** also stops and then restarts the node agent. Servers that were stopped when you clicked **Restart all Servers on Node** remain stopped.

Tip: The node agent for the node must be processing to restart application servers on the node.

- Stop the processing of a node agent. On the Node Agents page, select the check box beside the node agent that you want to stop processing; click **Stop**.

Results

Depending on the steps that you completed, you have viewed information about a node agent, stopped and started the processing of a node agent, and stopped and restarted application servers on the node that is managed by the node agent.

What to do next

You can administer other aspects of the Network Deployment environment, such as the deployment manager, nodes, and cells.

Node agent collection

Use this page to view information about node agents. Node agents are administrative agents that monitor application servers on a host system and route administrative requests to servers. A node agent is the running server that represents a node in a Network Deployment environment.

To view this administrative console page, click **System Administration > Node agents** .

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the addNode command and the startNode command.

Name

Specifies a logical name for the node agent server.

Node

Specifies a name for the node. The node name is unique within the cell.

A node name usually is identical to the host name for the computer. That is, a node usually corresponds to a physical computer system with a distinct IP host address.

However, the node name is a purely logical name for a group of servers. You can name the node anything you please. The node name does not have to be the host name.

Version

Specifies the product version of the node.

The product version is the version of a WebSphere Application Server node agent and Application Servers that run on the node.



Host Name

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the node agent.

Status

Indicates whether the node agent server is started or stopped.

Note that

	Started	The node agent is running.
	Stopped	The node agent is not running.

Node agent server settings

Use this page to view information about and to configure a node agent. A node agent coordinates administrative requests and event notifications among servers on a machine. A node agent is the running server that represents a node in a Network Deployment environment.

To view this administrative console page, click **System Administration > Node Agents > *node_agent_name***.

A node agent must be started on each node for the deployment manager node to collect and control servers that are configured on that node. If you use configuration synchronization support, a node agent coordinates with the deployment manager server to synchronize the configuration data of the node with the master copy that the deployment manager manages.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the `addNode` command and the `startNode` command.

The Runtime tab displays only when a node agent runs.

Name:

Specifies a logical name for the node agent server.

Data type String

Node:

Specifies the name of the node for the node agent server.

Data type String

Short name:

Specifies the short name of the node agent server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF) (for example, Resource Access Control Facility (RACF)), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeric.

The system assigns a cell-unique, default short name.

Unique ID:

Specifies the unique ID of this node agent server.

The unique ID property is read only. The system automatically generates the value.

Run in 64 bit JVM mode:

Specifies whether to run the node agent in 64-bit addressing mode to obtain more than the virtual memory that is available to the node agent when the node agent runs in 31-bit addressing mode, which is 2 gigabytes (GB).

Note: You should eventually convert all of your servers to run in 64-bit addressing mode because support for running servers in 31-bit addressing mode is deprecated.

Default true (checked)

Start components as needed: Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID:

Specifies a string identifying the process.

Data type String

Cell Name:

Specifies the name of the cell for the node agent server.

Data type String
Default *host_name*Network

Node Name:

Specifies the name of the node for the node agent server.

Data type String

State:

Indicates whether the node agent server is started or stopped.

Data type String
Default Started

Administrative agent

An administrative agent provides a single interface to administer multiple unfederated application server nodes in environments such as development, unit test or that portion of a server farm that resides on a single machine.

The administrative agent and application servers must be on the same sysplex, but you can connect to the sysplex from a browser or the wsadmin tool on another machine.

Multiple customers administer application servers in their development, test, and production environments by federating the application server nodes into a cell and administering the application servers from the deployment manager. However, if you have development and unit test environments, then you might prefer to run application servers whose nodes have not been federated. These application servers have some administrative disadvantages. The application servers lack a common administrative interface. Remote administration is limited to installing applications and changing application server configurations. As an alternative, you can register these application servers with an administrative agent to administer application servers from a single interface and to more fully administer application servers remotely.

You can register an application server node with the administrative agent or federate the node with a deployment manager, but not both.

An administrative agent can monitor and control multiple application servers on one or more nodes. Use the application servers only to run your applications. By using a single interface to administer your application servers, you reduce the overhead of running administrative services in every application server.

You can use the administrative agent to remotely install applications on application servers, change application server configurations, stop and restart application servers, and create additional application servers.

You can start the administrative agent on any logical partitions (LPAR) if the administrative agent configuration is on a shared file system.

Administering nodes using the administrative agent

You can configure an administrative agent, view or change unfederated application server nodes registered to the administrative agent, and view or change job manager configurations for a registered node. An administrative agent provides a single interface to administer application servers in, development, unit test, or server farm environments, for example.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

Note: If the administrative agent resides on a DMZ image, do not change the bootstrap port. Even though the administrative agent does not open the bootstrap port, the control region needs the bootstrap port definition to communicate with the servant region of the administrative agent.

Before you use the administrative agent, install the core product files, create an administrative agent profile, and start the administrative agent. Use the `registerNode` command to register at least one application server node with the administrative agent.

About this task

Note: The administrative agent provides a single interface to administer multiple unfederated application server nodes in, for example, development, unit test, or server farm environments. By using a single interface to administer your application servers, you reduce the overhead of running administrative services in every application server.

You can use the administrative console of the administrative agent to configure the administrative agent, view and change properties for nodes registered to the administrative agent, register and unregister application server nodes with job managers, and view and change job manager configurations for a registered node. A job manager allows you to asynchronously submit and administer jobs for a node registered to the administrative agent when the node is also registered to the job manager. Read the section on planning the installation for topologies that include administrative agents and job managers.

- View and change properties for the administrative agent.
 1. Click **System Administration > Administrative agent** from the navigation of the administrative agent administrative console.
 - Optionally click **Register with Job manager** to register an application server node with a job manager.
 - Optionally click **Unregister from a Job Manager** to unregister an application server node from a job manager.
 - Optionally view the administrative agent properties on the Configuration tab and the Runtime tab.
 - Optionally select **Start components as needed** on the Configuration tab. Click **Apply**, and then click **OK**.

Selecting the setting allows administrative agent components to start dynamically as needed for applications.
- View and change properties for a node registered to the administrative agent.
 1. Click **System Administration > Administrative agent > Nodes**.

You can view the nodes registered to the administrative agent.
 2. Click **System Administration > Administrative agent > Nodes > *node_name***.
 - Optionally select Poll jobs from job manager to have the administrative agent retrieve jobs from the job manager for this node.
 - To change other properties for the node, click the links under Additional Properties.
- View and change job manager configurations for a registered node.
 1. Click **System Administration > Administrative agent > Nodes > *node_name* > Job managers** to view the job managers to which the node is registered.
 2. Click **System Administration > Administrative agent > Nodes > *node_name* > Job managers > *job_manager_UUID*** to change job manager-related properties for the registered node.
 - Optionally change the polling interval by entering an integer value.

The administrative agent uses the polling interval to check for jobs from the job manager for this registered node.
 - Optionally change the Web address of the job manager that the administrative agent polls for this registered node.

Results

Depending on the tasks that you completed, you might have configured the administrative agent, registered and unregistered application server nodes with job managers, viewed or changed properties for a node registered to the administrative agent, or viewed and changed the job manager configuration for a registered node.

What to do next

You can continue to administer registered nodes from the administrative agent. You can further configure the administrative agent using the links on the configuration tab of the administrative agent panel. You can register more nodes with the administrative agent using the `registerNode` command. You can deregister nodes from the administrative agent using the `deregisterNode` command. You can register and unregister nodes with a job manager.

Administrative agent settings

This panel allows you to configure the administrative agent and view its properties.

To view this administrative console page, click **System Administration > Administrative agent**.

Name

Specifies the administrative agent server name. The name is read-only.

Node

Specifies a name for the administrative agent node. The node name is unique within the cell. The node name is read-only.

By default, a node name is the hostname appended with Node01. For example, a node on a computer with the host name of MyComputer is named MyComputerNode01 by default.

However, the node name is a purely logical name for a group of servers. The node name does not have to contain the host name.

Short name

Specifies the short name of the administrative agent server.

The server short name must be unique within a cell. The short name identifies the server to the native facilities of the operating system, such as workload manager (WLM), Automatic Restart Manager, System Authorization Facility (SAF), for example, Resource Access Control Facility (RACF), started task control, and others.

The name can be 1-8 alphanumeric or national language characters and cannot start with a numeral.

The system assigns a cell-unique, default short name.

Data type String

Unique ID

Specifies the unique ID of this administrative agent server.

The unique ID property is read-only. The system automatically generates the value.

Data type String

Start components as needed

Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID

Specifies the read-only process ID of the administrative agent.

Cell name

Specifies the read-only cell name of the administrative agent.

Node name

Specifies the read-only node name of the administrative agent.

State

Specifies the read-only state of the administrative agent, such as started or stopped.

Nodes collection for the administrative agent

This panel allows you to view the application server nodes that are registered to the administrative agent. The administrative agent provides a single interface to the registered nodes.

To view this administrative console page, click **System Administration > Administrative agent > Nodes**.

Application server nodes already registered to an administrative agent can also be registered to job managers. Job managers allow you to asynchronously submit and administer jobs for large numbers of unfederated application servers and deployment managers over a geographically dispersed area.

Register with Job manager

Allows you to register a node with the job manager

Unregister from a Job Manager

Allows you to unregister a node from the job manager

Name

Specifies a name for an application server node that is registered to the administrative agent. The name is read-only.

Unique ID

Specifies the unique ID of this application server.

The unique ID property is read-only. The system automatically generates the value.

Data type String

Registered nodes settings

This panel allows you to view properties for a node registered to the administrative agent. The properties are name, unique ID, and poll jobs from job manager.

To view this administrative console page, click **System Administration > Administrative agent > Nodes > *node_name***.

Name: Specifies the name of an application server registered to the administrative agent. The name is read-only.

Unique ID:

Specifies the unique ID of this application server.

The unique ID property is read only. The system automatically generates the value.

Data type String

Poll jobs from job manager: Select the option so that the administrative agent retrieves jobs from the job manager for this node. The jobs start when the administrative agent retrieves them. The polling interval is defined on the Job manager panel of the administrative console for the administrative agent and controls how often the administrative agent checks for new jobs. The default is to retrieve the jobs.

Register or unregister with job manager

This panel allows you to either register a node to a job manager or unregister a node from a job manager. The node can be a deployment manager or a node registered to an administrative agent.

You can use the administrative agent administrative console to register or unregister a node with the job manager. The node that you register with the job manager is already registered to the administrative agent. For example, click **System administration > Administrative agent > Nodes**. Select **Register with Job Manager** to register a node to a job manager or **Unregister from a Job Manager** to unregister a node from a job manager.

You can use the deployment manager administrative console to register or unregister a deployment manager with the job manager. Click **System administration > Deployment manager**. Under Additional properties, click **Job Managers > Register/unregister with job manager**.

Managed node name

A required setting that specifies the name of the managed node. For a deployment manager, the managed node name is the name of the deployment manager node, usually `host_nameCellManager01`.

Alias

An optional setting that specifies the alias of the managed node to enroll. Specify an alias if the managed node name is in use by another node.

Host name

An optional setting that specifies the host name to use to identify the job manager. The default value is `localhost`.

Port

An optional setting that specifies the job manager administrative console port number. If security is enabled, use the secure port number. If security is disabled, use the unsecure port number. The default secure port number is 9943, and the default unsecure port number is 9960. If no port number is specified, 9943 is used.

User name

Specifies the user name to log into the job manager when security is enabled.

Password

Specifies the password to log into the job manager. This setting is required when the user name setting is required.

Confirm password

Specifies the password a second time. This setting is required when a user name and a password are required.

Job managers collection

This panel allows you to view the job managers to which this node is registered. The job managers allow you to asynchronously submit and administer jobs, such as manage applications, for this node.

To view this administrative console page, click **System Administration > Administrative agent > Nodes > *node_name* > Job managers**.

UUID

Specifies the Universal Unique Identifier (UUID), which uniquely identifies the job manager to which the administrative agent connects.

URL

Specifies the Web address of the job manager to which the node is registered.

Job manager settings

This panel allows you to view the Universal Unique Identifier (UUID) of the job manager, specify the polling interval to check for jobs on the job manager, and specify the Web address of the job manager.

To view this administrative console page, click **System Administration > Administrative agent > Nodes > *node_name* > Job managers > *job_manager_UUID*** .

Job Manager UUID: Specifies the Universal Unique Identifier (UUID), which uniquely identifies the job manager to which the administrative agent connects. The UUID is read-only.

Polling interval:

Specifies in seconds the time that elapses during a polling interval.

The administrative agent uses the polling interval to determine how often to poll a job manager for new jobs for the registered node. The default value is 30 seconds.

URL:

Specifies the Web address that the administrative agent uses to connect to the job manager.

The formats are `http://host:port/otis/OMADMServlet` or `https://host:port/otis/OMADMServlet`, where *host* is the host name of the job manager, and *port* is the port of the job manager.

Job manager

In a flexible management environment, a job manager allows you to submit administrative jobs asynchronously for application servers registered to administrative agents and for deployment managers. You can submit these jobs to a large number of servers over a geographically dispersed area.

You can make both application server nodes that are registered to administrative agents and deployment managers known to the job manager through a registration process. After you register application server nodes and deployment managers with the job manager, you can queue administrative jobs directed at the application server nodes or deployment managers through the job manager.

To register application server nodes and deployment managers with the job manager, use the `wsadmin registerWithJobManager` command. The command is in the `ManagedNodeAgent` command group.

Use the job manager to asynchronously administer job submissions. You can complete the following tasks:

- Set the job submission to take effect at a specified time.
- Set the job submission to expire at a specified time.
- Specify that the job submission occur at a specified time interval.
- Notify the administrator through e-mail that the job has completed.

Nodes in terms of the job manager are the application server nodes and deployment managers registered to the job manager. Groups of nodes are those groups that you create so that you can make job submission easier. You can submit a job for a group of nodes instead of entering multiple node names for a job.

Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application management, server management, and node management. However, with the job manager, you can aggregate tasks and perform those tasks across multiple application servers or deployment managers.

The following hypothetical company environments are examples of situations where a job manager is useful:

Branch office environment

A business has a thousand stores geographically dispersed across the continent. Each store contains either a few application servers, or a small Network Deployment cell consisting of two or three machines. Each store is managed locally for daily operations. However, each store is also connected to the data center at the company headquarters, potentially thousands of miles away. Some connections to the headquarters are at modem speeds. The headquarters uses the job manager to periodically submit administrative jobs for the stores.

Environment consisting of hundreds of application servers

An administrator sets up hundreds of low-cost machines running identical clones of an application server. Each application server node, which is registered with an administrative agent, is registered with the job manager. The administrator uses the job manager to aggregate administration commands across all the application servers, for example, to create a new server, or to install or update an application.

Environment consisting of dozens of deployment manager cells

An administrator sets up hundreds of application servers, which are divided into thirty different groups. Each group is configured within a cell. The cells are geographically distributed over five regions, consisting of three to seven cells per region. Each cell is used to support one to fifteen member institutions, with a total of 230 institutions supported. Each cell contains approximately thirty applications, each running on a cluster of two for failover purposes, resulting in a total of 1800 application servers. The administrator uses the job manager to aggregate administration commands across all the cells, for example, to start and stop servers, or to install or update an application.

Administering nodes using the job manager

In a flexible management environment, you can asynchronously submit and administer jobs for large numbers of unfederated application servers and deployment managers over a geographically dispersed area. At the remote machines, you can use jobs to manage applications, modify the product configuration, or do general purpose tasks such as run a script.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

Before you use the job manager, install the core product files, create a job manager management profile, and start the job manager. To administer unfederated nodes using the job manager, register the nodes with the administrative agent before you register the nodes with the job manager. To register unfederated nodes and deployment managers with the job manager, use the `wsadmin registerWithJobManager` command. The command is in the `ManagedNodeAgent` command group.

About this task

Note: In a flexible management environment, the job manager allows you to asynchronously submit and administer jobs for large numbers of unfederated application servers and deployment managers over a geographically dispersed area. Many of the management tasks that you can perform with the job manager are tasks that you can already perform with the product, such as application

management, server management, and node management. However, with the job manager, you can aggregate the tasks and perform the tasks across multiple application servers or deployment managers.

You can submit jobs for groups of nodes that you define or for individual nodes. After you submit a job, you can check the job status, check the status of nodes, and check the status of node resources. You can view node resources for nodes and groups of nodes that you administer. You can configure the job manager and view its properties. Read the section on planning the installation for topologies that include administrative agents and job managers.

- **Submit a job.**

You can submit jobs to remote nodes to manage applications, modify the product configuration on remote machines, or do general purpose tasks such as run a script. You can specify when the jobs start, whether they are recurring, and when they are no longer available for submission.

- **Check the status of a job.**

You can check the status of jobs, the status of jobs at their nodes, and the job history of nodes. You can suspend, resume, or delete jobs on the Job status collection panel.

- **Administer nodes of the job manager.**

You can view nodes with their version numbers based on the results of the Find option and view node resources for nodes that you select. You can also view the properties and property values for a particular node.

- **Administer node resources of the job manager.**

You can view server, application, node, and cluster resources that are associated with nodes and groups of nodes registered to the job manager. You can also view the status of specific resources at each node and view properties for a particular node resource as a name-value pair.

- **Administer groups of nodes for the job manager.**

You can create, modify, delete, and view groups of nodes. Groups of nodes make job submission simpler because you can submit a job for a group of nodes instead of entering multiple node names for a job submission.

- **Configure job managers.**

You can specify settings such as the default job expiration, the job manager Web address, and the mail provider Java Naming and Directory Interface (JNDI) name for the job manager. You can view job manager properties such as the process ID and the state of the job manager.

Results

Depending on the tasks that you completed, you might have submitted jobs, checked the status of jobs, viewed nodes and node resources, or administered groups of nodes.

What to do next

You can continue to administer jobs as described in the procedure. You can register nodes using the `wsadmin registerWithJobManager` command, or unregister nodes using the `wsadmin unregisterWithJobManager` command. Both commands are in the `ManagedNodeAgent` command group. You can stop and restart the job manager.

Submitting a job

In a flexible management environment, you can submit jobs to remote nodes to manage applications, modify the product configuration on remote machines, or do a general purpose task such as run a script. You can specify when the jobs start, whether they are recurring, and when they expire.

Before you begin

Before you can submit a job, you must have registered at least one node with the job manager. A node can be an application server node that was first registered with an administrative agent or a deployment manager node.

Your ID must be authorized for the administrator role or the operator role to submit jobs.

You can simplify administration of a large number of nodes by submitting jobs against groups of nodes. Each group of nodes represents a group of nodes. Before you can submit a job for a group of nodes, you must have created the group of nodes.

About this task

You can use the administrative console of the job manager to submit jobs to do tasks such as manage applications, modify the product configuration on remote workstations, or do general purpose tasks such as run a script. To complete the job submission, choose the type of job, choose the nodes on which you want the job to run, specify the job parameters that are specific to the job type, schedule the job, review the summary, and submit the job.

1. Click **Jobs > Submit** from the navigation tree of the job manager administrative console.
2. Choose the job type.

- a. Select the job type from the list.

The list of job types varies based on the nodes that you have registered with the job manager. The values displayed in the list are retrieved from the `getJobTypes` and `getJobTypeMetadata` commands of the `AdminTask` object. You can have job types that manage applications, modify the product configuration on remote machines, or do general purpose tasks such as run a script.

- b. Optionally specify a description of the job.

The description is a string that can be up to 256 characters. The default description is the job type. You can change or add to the default description. The description is useful when using the `Find` option to view existing jobs.

- c. Click **Next**.

3. Choose the job targets.

You are determining the nodes on which you want the job to run.

- a. Select a group of nodes from the list, or select **Node name**.

Only groups of nodes that are valid for the job type that you selected are displayed in the list of groups of nodes.

- b. If you selected **Node name**, then enter a node name, and click **Add**, or compile a list of nodes by using the **Find** option.

Node name that you enter

If you enter a node name, it must be a node that has been registered to the job manager. The node name is validated when you click **Next**.

List of node names

- 1) Click **Find**.

The Find nodes panel is displayed.

- 2) If you want to run the Find operation on specific keywords, specify a valid operator and a text string.

The list of keywords is dynamic. Valid operators are `=` (equal to), `!=` (not equal to), `is null`, and `is not null`. The text string can be complete or partial and can contain an asterisk (*) to include variable or unknown characters.

- 3) Click **Find**.
The results are displayed in the Available nodes list and are selected.
- 4) Move nodes between the Available nodes list and the Chosen nodes list.
 - To move specific nodes from the Available nodes list to the Chosen nodes list, select nodes in the Available nodes list and click >.
 - To move specific nodes from the Chosen nodes list to the Available nodes list, select nodes in the Chosen nodes list and click <.
- 5) After you have a list of the desired nodes in the Chosen nodes list, click **OK**.
The nodes display on the Choose job targets panel.

c. Click **Next**.

4. Specify the job parameters.

The list of job parameters is dynamic and based on the job type. For example, if the job type is to install an application, you would specify the application name, the location of the application to install, and optionally the name of the server where the system installs the application. The following table describes the types of parameters.

Parameter Type	Description
String	You can enter text for the appropriate parameters. The text is not validated until the job is submitted.
Node resource	You can select a node resource. The Find option is available for you to search for the resource, depending on the job type that you selected in the first step.

- a. Optionally click **Find** if it is available.
The Find node resources panel is displayed.
- b. If you want to run the Find operation on specific keywords, specify a valid operator and a text string.
The list of keywords is dynamic. Valid operators are = (equal to), != (not equal to), is null, and is not null. The text string can be complete or partial and can contain an asterisk (*) to include variable or unknown characters.
- c. Click **Find**.
The results are displayed in the Available resources common to all selected endpoints list.
- d. Click **OK** to save the results and return to the panel on specifying job parameters.
- e. Click **Next**.

5. Schedule the job.

The times and dates that you specify are relative to the job manager.

- a. Optionally specify one or more e-mail addresses where notifications are sent when the job is done.
If you specify multiple e-mail addresses, separate them by commas. The e-mail addresses are saved in your console preferences. Each e-mail address is validated for format errors.
- b. Select when the job is available for submission.
You can submit the job to be available now, or specify a time and date that the job is retrieved from the job manager.
- c. Select the job expiration.
The job expiration is the time at which the job will no longer be available for nodes to run. You can use the default expiration, specify a time and date for the job expiration, or specify an amount of time in which the job expires. The default expiration is defined on the Job manager configuration panel.
- d. Optionally specify a recurring interval for the job, a start date and time for the interval, and an end date and time for the interval.

- e. Click **Next**.
6. Review the summary, and submit the job.
 - a. If you want to make changes to the options, click **Previous** until you reach the panel that you want to change. Make the correction on that panel, and then proceed through the panels until you reach the Summary and submit panel.
 - b. When you are satisfied with the options, click **Finish** to submit the job.

The Job status collection panel is displayed where only the status for the job that you submitted is displayed.

Results

Depending on the type of job that you selected, you have submitted a job to manage applications, modify the product configuration on remote machines, or do a general purpose task such as run a script.

What to do next

You can check the job status or submit other jobs.

Find node resources

This panel allows you to find node resources for resource job parameters when you submit a job through the job submission wizard. This panel is used for job types such as start server, stop server, and start cluster.

To access this panel, first click **Jobs > Submit**. The job type that you select in step 1 of the Job submission wizard affects whether you can access the Find node resources panel in step 3. For example, if you select the job type, start server, in step 1, then in step 3 you can select **Find** to access the Find node resources panel.

Find:

You can use the Find option to determine the node resources to display. The Find results are displayed in **Available resources common to all selected nodes**, after you click **Find**. Click **Reset** to clear the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
<p>Except for the maximum results parameter, this column lists parameters that define node resources. The node name, job type, and unique identifier parameters are always available.</p>	<ul style="list-style-type: none"> • All parameters except type and maximum results: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. • Type: They type is preselected to the type of resource option that you need for job submission. You cannot change the type. The list contains one or more values of Server, Application, or Cluster. 	<ul style="list-style-type: none"> • All parameters except type and maximum results: Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the node resource parameter to Server* finds all server type resources that start with Server. You can search for an exact match for multiple items by including comma-separated items. For example, you can search on two resource names by entering server1, server2. When you search for more than one item, you cannot use the asterisk. • Node names are included based on your step 2 choices. • Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the resources are server1, server2, and server3, you can enter server or server* for column 3 and = for column 2 of the resource name parameter.

Available resources common to all selected nodes: Specifies the results of the find operation. If no choices are listed or you do not see the resource you want, you can search again. If your job is targeted to multiple nodes, only those resources that are present on all nodes will be listed. If at least one resource is listed, you can select one resource and click OK to return it to the parameter on the previous console panel.

Checking job status

In a flexible management environment, you can check the overall status of jobs, the status of specific job nodes, and the job history of nodes. You can suspend, resume, or delete jobs on the Job Status collection panel.

Before you begin

Before you can check the job status, you must have registered at least one node with the job manager and submitted a job for the node.

To suspend a started job, resume a suspended job, or delete selected jobs, your ID must be authorized for the operator role.

About this task

After you finish submitting a job through the Job Submission wizard, the Job status collection panel is displayed. When this panel is displayed, the panel contains only the job that you submitted along with its status information. You can check the status of the job, the status of job at its nodes, and the job history of the nodes.

If you access the Job status collection panel by selecting **Jobs > Status** in the job manager administrative console navigation, you can use the Find option to limit the number of jobs that are displayed based on the criteria you specify. The first time you access the Job status collection, no jobs are listed. You must enter parameters for the Find option to obtain a list of jobs based on the parameter information that you provide. The next time you select **Jobs > Status**, a list of jobs is displayed based on the parameters you last specified on the Find option for this job manager administrative console panel. You can then optionally modify the Find option criteria to display a different set of jobs. After at least one job displays, you can check the status of the displayed jobs, the status of specific job nodes, and the job history for nodes of a particular job.

- Optionally use the Find option to display a set of jobs.

If no jobs are displayed, you must use the Find option to display jobs based on the parameter information that you enter.

1. Click **Jobs > Status** in the job manager administrative console navigation.
2. For the parameters on which you want to do a Find operation, specify an operator and a text string.
3. Click **Find**.

The list of jobs along with their status information are in the collection table.

- Check the status of a job at its nodes.

1. Select **Jobs > Status** in the console navigation to access the Job status collection panel if you did not get to the panel as a result of a job submission.
2. Select either a job from the Job ID column or a number on the graph in the Status Summary column for a particular job. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the nodes in the job.
3. Optionally use the Find option to display the status of specific job nodes based on the parameter information that you enter.
 - a. If you want to run the Find operation on specific parameters, specify an operator and a text string as appropriate.
 - b. Click **Find**.

A list of nodes for the job, along with the status for each node, are displayed on the Job status detail panel.

- Check the job history of nodes.

1. Select **Jobs > Status** in the job manager administrative console navigation to access the Job status collection panel if you did not get to the panel as a result of a job submission.
2. Select either a job from the Job ID column or a number on the graph in the Status Summary column for a particular job. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the nodes in the job.
3. On the Job status detail panel, click a link in the Status column for one of the nodes names.
4. Optionally use the Find option to display job history based on the parameter information that you enter.
 - a. If you want to run the Find operation on specific parameters, specify an operator and a text string as appropriate.
 - b. Click **Find**.

The status of the job for the node is displayed on the Job status history panel.

- Suspend a job.

1. Select **Jobs > Status** in the job manager administrative console navigation to access the Job Status Collection if you did not get to the panel as a result of a job submission.
 2. Select the check box next to a job with an active or pending state.
 3. Click **Suspend**.
- Resume a job.
 1. Select **Jobs > Status** in the job manager administrative console navigation to access the Job status collection panel if you did not get to the panel as a result of a job submission.
 2. Select the check box next to a job whose state is Suspended.
 3. Click **Resume**.
 - Delete a job.
 1. Select **Jobs > Status** in the job manager administrative console navigation to access the Job status collection panel if you did not get to the panel as a result of a job submission.
 2. Select the check box next to the job that you want to delete.
 3. Click **Delete**.

Results

You might have run a Find operation to display job status based on criteria that you specify, checked the status of jobs at their nodes, checked the jobs history of nodes, suspended a job, resumed a job, or deleted a job.

What to do next

You can continue to check job status and do other job management tasks such as submit other jobs, create node groups for job submission, view node resources, or view nodes.

Job status collection

This panel displays information about submitted jobs, including the job ID, description, state, activation time, expiration time, and status summary. Jobs are submitted to administer nodes that have been registered with the job manager.

Deployment manager nodes and unfederated application server nodes that have been registered with an administrative agent are eligible to be registered with the job manager. The job manager can administer the entire deployment manager cell, not just the deployment manager.

This console panel is navigated to in two ways.

- When the job submission wizard completes, this console panel is displayed only with the job that you just submitted.
- Click **Jobs > Status** to view the status of any your submitted jobs.

To suspend a started job, resume a suspended job, or delete selected jobs, your ID must be authorized for the operator role.

Use one of the following buttons to suspend, resume, or delete a job.

Button	Description
Suspend	Specifies that an administrative agent or deployment manager can no longer retrieve the job.
Resume	Specifies that an administrative agent or a deployment manager can retrieve the job.

Button	Description
Delete	Specifies that a job and all its history are removed and no longer available. When you click Delete , you are given an opportunity to confirm or cancel the delete operation.

Find:

When you click **Jobs > Status**, you can use the Find option to limit the submitted jobs to display. The Find results are displayed in the table that follows the Find and Preference options after you click **Find**. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
Except for the maximum results parameter, this column lists parameters that define a job.	<ul style="list-style-type: none"> Job ID, state, node name, group of nodes, and description parameters: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. Activation time and the expiration time parameters: Valid operators are >=, and <=. 	<ul style="list-style-type: none"> All parameters except maximum results: Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the description to inventory* finds all jobs with a description that starts with inventory. You cannot use the asterisk in the job ID field or the fields related to time. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two job IDs by entering 119489625729609463, 119489651801509472. When you search on more than one item, you cannot use the asterisk. Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the nodes are EastCoast1, EastCoast2, WestCoast1, and WestCoast2, you can enter East or East* for column 3 and = for column 2 of the node parameter. Job status is displayed for the jobs that include the EastCoast1 and EastCoast2 nodes.

Job ID: Specifies the job number of the job that you submitted.

Description: Specifies the description that you entered when you submitted the job.

State:

Specifies whether the state of the job is Active, Pending, Expired, or Suspended.

State	Description
Active	The job is activated and not expired. The job does not have to be running to be active.
Pending	The job has been submitted, but has not been activated.

State	Description
Expired	The expiration time that you specified during job submission was reached before the job was started. A job does not start on a node after the expiration time is reached. If the expiration time is reached while the job is running on any of its nodes, the job continues on those nodes.
Suspended	The job has been suspended. If the suspension occurs while the job is running on a node, the job continues on that node. If the suspended job has not started when the suspension occurs, the job will not start on any of its nodes unless someone resumes the job. You can suspend a job from this console panel by clicking Suspend . You can resume a job from this console panel by clicking Resume .

Activation time:

Specifies the activation time that you entered when you submitted the job or the actual time when the job was submitted if you did not specify an activation time.

The activation time is the time the job is available to run on the targeted nodes or groups of nodes.

Expiration time:

Specifies the time the job is to expire. If the job has not started by the expiration time, the job will not start and the state will change to `Expired`. If the expiration time occurs while the job is running on a node, the job continues on that node.

When you submit the job, you can set the expiration date and time, choose the default expiration time option, or specify the amount of time until the job expires.

Status summary:

Graphically provides an overview of how the job is running at its nodes. The graph is divided in up to four sections, indicating success, partial success, failure, or other, in that order, of the nodes in the job.

Status	Description
Success	Indicates the number of nodes on which the job completed successfully.
Partial success	Indicates the number of nodes on which the job partially completed. Partial success can occur, for example, when a node represents multiple servers, and only some of the servers on the node complete successfully.
Failed	Indicates the number of nodes on which the job failed to complete.
Other	Indicates the number of nodes on which the job has some other status than success, partial success, or failure. A status of other can include nodes on which the job is currently running, or nodes on which the job has not started.

Job status detail settings:

This panel displays the job status such as success, partial success, or failed, at each node of the job. Job information, including the job ID, description, activation time, and expiration time, is also displayed.

To view the status of all the nodes for a particular job, click **Jobs > Status > job ID**.

To view only the successful nodes, only the failed nodes or only nodes with a status of other for a particular job, click **Jobs > Status > number for the successful, failed or other nodes in the status summary**.

Find:

When you click **Jobs > Status > job ID**, you can use the Find option to limit the nodes to display. The Find results are displayed in the table that follows the Find option after you click **Find**. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
This column lists the node name parameter, the status parameter and the maximum results parameter.	<ul style="list-style-type: none"> Job ID parameter: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. Status parameter: Valid operators for the find operation are Succeeded, Failed, Rejected, In progress, Distributed, and Not attempted. In progress is the same as ASYNC_IN_PROGRESS in the status column. Not attempted is the same as NOT_ATTEMPTED in the status column. 	<ul style="list-style-type: none"> For the node name parameter, specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the node name to AppSrv* finds all jobs with a node name that starts with AppSrv. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two node names by entering AppSrv01, AppSrv02. Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the nodes are EastCoast1, EastCoast2, WestCoast1, and WestCoast2, you can enter East or East* for column 3 and = for column 2 of the node parameter. Job status is displayed for the jobs that include the EastCoast1 and EastCoast2 nodes.

Job ID: Specifies the job number of the job that you submitted.

Description: Specifies the description that you entered when you submitted the job.

Activation time:

The activation time is the time that the job is available to start, but not necessarily the time that the job actually starts.

You entered the activation time when you submitted the job.

Expiration time:

Specifies the expiration time that you entered when you submitted the job.

If the job has not started by the expiration time, the job will not start.

Node name: Specifies the name of each node that is included in the job.

Status:

Specifies the status of the job at its nodes.

Status	Description
Succeeded	The job completed successfully on the node.
Partially succeeded	The job partially completed on the node. Partial success can occur, for example, when a node represents multiple servers, and only some of the servers on the node complete successfully.
Failed	The running of the job on the node failed.
Not attempted	The agent for the node has not received the job.
Distributed	The agent for the node has received the job, but the job has not completed.
In progress	The agent for the node has received the job and is running the job concurrent with other jobs for other nodes that belong to the agent.
Rejected	The agent rejected the job because the agent does not support the job type. The rejection can happen if a new node is added to the job group after the job is submitted.

Job status history:

This panel displays the job status of the node at various stages of the job for the node.

To view this administrative console panel, click **Jobs > Status > job ID > node name**.

The **Previous records** button and **Next records** button allow you to scroll backwards and forwards through the list of records if you are not viewing all records in the job history for the node. More records can be generated as the job progresses. These records are retrieved and included in the job history records as you click the **Previous records** button and the **Next records** button.

Find:

When you click **Jobs > Status > job ID > node name**, you can use the Find option to limit the submitted jobs to display. The Find results are displayed in the table that follows the Find option after you click **Find**. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
Except for the maximum results parameter, this column lists parameters that define a job.	<ul style="list-style-type: none"> Job ID and node name parameters: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. Time stamp: Valid operators are >=, and <=. 	<ul style="list-style-type: none"> For the node name parameter, specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the node name to AppSrv* finds all jobs with a node name that starts with AppSrv. You cannot use the asterisk in the job ID field or the time stamp field. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two job IDs by entering 119489625729609463, 119489651801509472. Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Time stamp: Specifies the date and time at which the status is logged for the job running at the node.

Status:

Specifies the status of the job running at the node for various stages of the job. The following table lists valid statuses with a description of each status.

Status	Description
Succeeded	The job completed successfully on the node.
Partially succeeded	The job partially completed on the node. Partial success can occur, for example, when a node represents multiple servers, and only some of the servers on the node complete successfully.
Failed	The running of the job on the node failed.
Not attempted	The agent for the node has not received the job.
Distributed	The agent for the node has received the job, but the job has not completed.
In progress	The agent for the node has received the job and is running the job concurrent with other jobs for other nodes that belong to the agent.
Rejected	The agent rejected the job because the agent does not support the job type. The rejection can happen if a new node is added to the job group after the job is submitted.

Message: Specifies error messages associated with the job when the status is Failed.

Administering nodes of the job manager

In a flexible management environment, you can view nodes with their version numbers based on the results of the Find option, and view node resources for nodes that you select. You can also view the properties and property values for a particular node.

Before you begin

Before you can view information about nodes, you must have registered at least one node with the job manager.

To display resources for selected nodes, your ID must be authorized for the monitor role.

About this task

The first time you access the Nodes collection panel, no nodes are listed. You must enter parameters for the Find option to obtain a list of nodes based on the parameter information that you provide. The next time you select **Jobs > Nodes**, a list of nodes are displayed based on the parameters you last specified on the Find option for this job manager administrative console panel. You can then optionally modify the Find option criteria to display a different set of nodes. After at least one node is displayed, you can view information about the nodes. You can display all nodes by setting the string for the node name on the Find option to * .

- Optionally use the Find option to display a set of nodes.

If no nodes are displayed, you must use the Find option to display nodes based on the parameter information that you enter.

1. Click **Jobs > Nodes** in the job manager administrative console navigation.
2. If you want to run the Find operation on specific parameters, specify a valid operator and a text string. You can use the asterisk (*) character on the node name, job type, and unique identifier parameters to represent unspecified characters in conjunction with the characters that you specify. The node Find option has some advanced Find options that you can view by selecting the plus (+) character. You can enter partial search strings for the advanced find options as well.
3. Click **Find**.

The list of nodes along with their version number is displayed in the collection table.

- Optionally display resources for selected nodes.

1. Click **Jobs > Nodes** in the job manager administrative console navigation.
2. Select the check box next to nodes for which you want to display resources.
3. Click **Display resources**.
4. Choose the type of resources to display.

The resources are displayed for the nodes that you selected.

- Optionally display properties and property values for a particular node by clicking **Jobs > Nodes > *managed_node_name*** in the job manager administrative console navigation.

Results

Depending on the tasks that you completed, you might have viewed nodes with their version numbers based on the results of the Find option, viewed node resources for nodes that you selected, or viewed the properties and property values for a particular node.

What to do next

You can continue to view nodes and do other job management tasks such as submit jobs, create node groups for job submission, and view node resources.

Nodes collection for Find results

This panel displays nodes with their version numbers based on the results of the Find option. You can additionally display node resources for nodes that you select.

To view this administrative console page, click **Jobs > Nodes**.

Find:

When you click **Jobs > Nodes**, you can use the Find option to determine the nodes to display. After you click **Find**, the results are displayed in the table. The table follows the Find and Preferences options. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
Except for the maximum results parameter, this column lists parameters that define nodes. The node name, job type, and unique identifier parameters are always available. The rest of the parameters are for the node properties, dynamic, and built at run time. The dynamic parameters are displayed when you click the Advanced find options link.	<ul style="list-style-type: none">All parameters except maximum results: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null.	<ul style="list-style-type: none">All parameters except maximum results and type: Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the node name parameter to Node* finds all jobs with a node name that starts with Node. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two node names by entering Node1, Node2. When you search on more than one item, you cannot use the asterisk.Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the nodes are AppSvr01, AppSvr02, AppSvr03, and Test01, you can enter App or App* for column 3 and = for column 2 of the node parameter. The node names displayed in the table are AppSvr01, AppSvr02, and AppSvr03.

Display resources: To display node resources of selected nodes, click **Display resources**. Your ID must be authorized for the monitor role. A dropdown list displays all available resources. Possible values are All, Applications, Servers, and Clusters. The choices in the list depend on the nodes registered with the job manager. For example, if you do not have a deployment manager registered to the job manager, clusters are not in the list.

Node name: Specifies the node names that are found as a result of the find option.

Version:

Specifies the product name and version number of the node.

The product version is the version of a WebSphere Application Server for nodes.

The base edition of WebSphere Application Server is listed in the version column as Base. The Network Deployment product is listed in the version column as ND.

The product in the version column indicates the product that you used to create the profile, not the type of profile that you installed. For example, if you use the Network Deployment product to install a profile type of application server, the version column indicates ND.

Node properties:

This panel allows you to view the properties and property values for a particular node.

To view this administrative console page, click **Jobs > Nodes > *managed_node_name***.

Node name: Specifies the node name that you selected from the nodes collection.

Name: Specifies the name of each property for the node. The list of names varies from one runtime to another and is read-only.

Value: Specifies a value of the specified name. The value is read-only.

Administering node resources of the job manager

In a flexible management environment, you can view server, application, and cluster resources associated with nodes and node groups registered to the job manager. You can also view the status of specific resources at each node, and view properties for a particular node resource as a name-value pair.

Before you begin

Before you can view information about node resources, you must have registered at least one node with the job manager.

About this task

The type of resources you can view depends on your topology. For example, you cannot view clusters if a deployment manager that you registered to the job manager does not have a defined cluster.

The first time you access the Node resource collection panel, no node resources are listed. You must enter parameters for the Find option to obtain a list of node resources based on the parameter information that you provide. The next time you select **Jobs > Node resources**, a list of node resources are displayed based on the parameters you last specified on the Find option for this console panel. You can then optionally modify the Find option criteria to display a different set of node resources. After at least one node resource displays, you can view information about the node resources.

- Optionally use the Find option to display a set of node resources.

If no node resources are displayed, you must use the Find option to display node resources based on the parameter information that you enter.

1. Click **Jobs > Node resources** in the console navigation.
2. Choose the resource type in the **Type** list.
3. If you want to do a Find operation on specific parameters, specify a valid operator and a text string.
4. Click **Find**.

The list of node resources along with the number of node resources for a given node resource in the list and the node for the resource are displayed.

- Optionally display the status of specific node resources for each node.

1. Click **Jobs > Node resources > *managed_resource*** in the job manager administrative console navigation

The resource ID, node name, and resource status are displayed.

- Optionally display the properties of a node resource by clicking **Jobs > Node resources > managed_resource > managed_resource_ID** in the job manager administrative console navigation.

Results

Depending on the tasks that you completed, you might have viewed server, application, and cluster resources associated with nodes and node groups registered to the job manager. You might have also viewed the status of specific resources at each node, and viewed properties for a particular node resource as a name-value pair.

What to do next

You can continue to view node resources and do other job management tasks such as submit jobs, create node groups for job submission, and view nodes.

Node resources collection

This panel displays server, application, node, and cluster resources associated with nodes and node groups registered to the job manager.

To view this administrative console panel, click **Jobs > Node resources**.

Find:

You can use the Find option to determine the resources to display. The Find results are displayed in the table that follows the Find and Preference options after you click **Find**. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
<p>Except for the maximum results parameter, this column lists parameters that define the resources of nodes. The node name and type parameters are always displayed. The rest of the resource parameters in the list vary from node to node.</p> <p>Node name Specifies the name of the node on which the resource resides.</p> <p>Type Specifies the type of resource. Options are server, application, and cluster.</p> <p>Status Specifies the status of the resource. Valid values for the status vary because different resources have different status. Examples of status for a server are started and stopped.</p> <p>Context Specifies topology information of the resource, such as cell/application.</p> <p>Resource name Specifies the name of a resource type. For example, a server resource could have a resource name of server1.</p> <p>Maximum results The maximum number of results that display after the find option completes.</p>	<ul style="list-style-type: none"> • Type: Valid values for the type of resource are server, application, and cluster. • Status, node name, context, and resource parameters: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. When you search on more than one item, you cannot use the asterisk. 	<ul style="list-style-type: none"> • All parameters except type and maximum results: Specifies the string or partial string of a parameter. • Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the resource names are server1, server2, application01, and application03, you can select an operator of != in column 2 for the resource name parameter, and a resource name of server or *erver in column 3, The results of the find operation display the application01 and application03 resource information.

Resources: Specifies the ID of the resource.

Number: Specifies the number of resources with this name registered with the job manager.

Node name: Specifies the name of the node for the resource. When the resource is defined on multiple nodes, the node name is displayed as multiple, and the number is greater than one. When you click on the resource name, you can see the details of the nodes where this resource resides.

Node resources for nodes collection:

This panel displays the status of specific resources at each node, and includes the resource ID, the name of the node, and the resource status.

To view this administrative console panel, click **Jobs > Node resources > resource_ID**.

Resource ID: Specifies a unique identifier for the resource in the form of context and values. For example, servers display as

server/*server_name* for nodes registered with an administrative agent

node/*node_name*/server/*server_name* for nodes federated in a registered deployment manager

Node name: Specifies the node on which the resource is located.

Status: Specifies the status of the resource. Valid values for the status varies because different resources have different status. Examples of status for a server are started and stopped.

Node resource properties:

This panel displays a read-only view of the properties for a particular node resource as a name-value pair. Updates that you make to the node resource properties must be done at the administrative agent.

To view this administrative console panel, click **Jobs > Node resources > resource_ID > resource_ID**.

Name:

Specifies the name of the property. The property name is unique.

Value:

Specifies the value paired with the specified name.

Administering groups of nodes for the job manager

In a flexible management environment, you can create, modify, delete, and view groups of nodes. Groups of nodes make job submission simpler because you can submit a job for a group of nodes instead of a entering multiple node names for a job submission.

Before you begin

Before you can add a node to a group of nodes, you must have registered at least one node with the job manager.

About this task

Groups of nodes are particularly useful if you submit multiple jobs to the same set of nodes.

The first time you access the Groups of nodes collection panel, no groups of nodes are listed. You must create at least one group. You must then enter parameters for the Find option to obtain a list of groups of nodes based on the parameter information that you provide. The next time you select **Jobs > Groups of nodes**, a list of groups of nodes are displayed based on the parameters you last specified on the Find option for this job manager administrative console panel. You can then optionally modify the Find option criteria to display a different set of groups of nodes. After at least one group of nodes is displayed, you can administer the groups of nodes by doing such tasks as adding and removing members for node groups, or deleting node groups.

- Create a group of nodes.
 1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
 2. Click **New**.
 3. Enter the name of the group of nodes.
 4. Optionally enter a description.
 5. Optionally add members to or remove members from the group of nodes.

Members are nodes. You can add members to the group or delete members from the group now or later. You can use the Add option, the Find option, or both options to add the members. Follow the steps on adding to or removing members from a group of nodes.

6. Click **Apply** to save the changes, and then click **OK** to return to the collection page.

- Optionally use the Find option to display groups of nodes.

If no groups of nodes are displayed, you must use the Find option to display groups of nodes based on the parameter information that you enter.

1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
2. Specify a valid operator and a text string.
3. Click **Find**.

- Optionally add or remove the members in a group of node.

You can add and remove members from the group of nodes. Members are nodes.

1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
2. Click one of the names of a group of nodes.
3. To add a node, use the Add option, the Find option, or both.
 - a. To use the Add option, type the name of the node in the **Member** list box, and then click **Add**.
 - b. Continue to type the name of a node, and then click **Add** until you have added all the members.
 - c. To use the Find option, click **Find**.
 - 1) Enter criteria for the Find option by adding text for one or more options. For example, specify the node name as test* or test*a.
 - 2) After getting the list of nodes in the **Chosen nodes** list, click **OK** to return the list to the Groups of nodes panel.

You can change the find criteria, and select the Find option multiple times to create the list you want.

4. To remove a node, select the node, and click **Remove**.
 5. Click **Apply**, and then click **OK**.
- Optionally delete one or more groups of nodes.
 1. Click **Jobs > Groups of nodes** in the job manager administrative console navigation.
 2. Select one or more groups of nodes.
 3. Click **Delete**.

Results

Depending on the tasks that you completed, you might have created a group of nodes, used the Find option to display groups of nodes, added or deleted members in the group of node, or deleted groups of nodes.

What to do next

You can continue to administer groups of nodes and do other job management tasks such as view nodes, submit jobs, and view node resources.

Groups of nodes collection

This panel allows you to create and view groups of nodes. Groups of nodes make job submission easier because you can submit a job for a group of nodes instead of a separate job for each node.

To view this administrative console page, click **Jobs > Groups of nodes**.

To create or delete a group, you must be authorized for the configurator role.

Find:

When you click **Jobs > Groups of nodes**, you can use the Find option to limit the groups of nodes to display. The Find results are displayed in the table that follows the Find and preferences options after you click **Find**. Clicking **Reset** clears the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
This column lists the group name parameter, the job type parameter, the description parameter, and the maximum results parameter.	<ul style="list-style-type: none">Group name parameter, job type parameter, and description parameter: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null.	<ul style="list-style-type: none">Group name parameter, job type parameter, and description parameter: Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the group name parameter to Region* finds all jobs with a group name that starts with Region. You can search on an exact match for multiple items by including the items to match separated by commas. For example, you can search on two group names by entering Region1, Region2. When you search on more than one item, you cannot use the asterisk.Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If you have already defined groups Region1, Region2, Center1, and Center2, you can enter Region or Region* for column 3 and = for column 2 of the group name parameter. When you click **Find**, only groups Region1 and Region2 will be displayed in the collection.

Group name: Specifies the name of the group.

Members: Specifies the number of members in the group. A member is a node.

Description: Specifies a description of the group.

Groups of nodes settings:

This panel allows you to set the name of a group of nodes, description, and group members. Groups of nodes make job submission easier because you can submit a job for a group of nodes instead of a separate job for each node.

To view this administrative console panel, click **Jobs > Groups of nodes > group_name**.

Group name: Specifies the name of the group. When you create a new group of nodes, the name is verified to be unique.

Data type	String
Maximum length	64 characters
Default	None

Description: Specifies an optional string that describes the group.

Data type	String
Maximum length	256 characters
Default	None

Member list: Specifies nodes to add to the group. You can either type the node name or use the **Find** option to add nodes to the list. The node must already exist. There are two types of nodes for the job manager. A node that is registered to an administrative agent and to the job manager is a node. A deployment manager that is registered to the job manager is a node.

Find nodes:

This panel allows you to build a list of nodes that you can use to choose the nodes on which you want the job to run. You can also find nodes to add to a group of nodes.

This console panel is navigated to in two ways.

- On step 2 of the Job submission wizard you can select **Find**.
- Click **Jobs > Groups of nodes > group_name > Find**.

Find:

When you click **Jobs > Nodes**, you can use the Find option to determine the nodes to display. The Find results are displayed in Available nodes after you click **Find**. Click **Reset** to clear the operators in column 2 and text in column 3.

Column 1	Column 2	Column 3
<p>Except for the maximum results parameter, this column lists parameters that define nodes. The node name, job type, and unique identifier parameters are always available. The remaining parameters are for the node properties, dynamic, and built at runtime. The dynamic parameters are displayed when you click Advanced find options.</p>	<ul style="list-style-type: none"> • All parameters except maximum results: Valid operators for the find operation are = (equal to), != (not equal to), is null, and is not null. 	<ul style="list-style-type: none"> • All parameters except maximum results: Specifies the string or partial string of a parameter. A partial string is designated using an asterisk (*). For example, setting the node name parameter to Node* finds all jobs with a node name that starts with Node. You can search for an exact match for multiple items by including comma-separated items. For example, you can search on two node names by entering Node1, Node2 When you search for more than one item, you cannot use the asterisk. • Maximum results: Specifies the number of records that the find operation displays. Enter a value between one and the maximum number of records that can be retrieved as defined in the job manager configuration.

Example: If the nodes are AppSvr01, AppSvr02, AppSvr03, and Test01, you can enter App or App* for column 3 and = for column 2 of the node parameter. The node names displayed in the table are AppSvr01, AppSvr02, and AppSvr03.

Available nodes: Specifies the initial results of the Find option. The nodes are not included in the list you are building until you use the right arrow to move nodes into the chosen nodes list. You can select multiple items in the list, and move them simultaneously.

Chosen nodes: Specifies the list of nodes to be included in the group of nodes that you are creating. The left arrow is used to move nodes into the available nodes list. You can select multiple items in the list, and move them simultaneously.

Configuring job managers

In a flexible management environment, you can specify settings such as the default job expiration, the job manager Web address, and the mail provider Java Naming and Directory Interface (JNDI) name for the job manager. You can view job manager properties such as the process ID and the state of the job manager.

Before you begin

Before you can configure the job manager, you must have started the job manager.

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

When you create a job manager profile, a job manager is created. You can run the job manager with its default settings. However, you can change the job manager configuration settings, such as the expiration time of jobs, the maximum number of database results to display, and so on.

1. Click **System administration > Job manager** from the navigation tree of the job manager administrative console to access the settings page for a job manager.
2. Configure the job manager by clicking a property and specifying settings.
 - a. Optionally change the default job expiration by specifying an integer value.
 - b. Optionally change the maximum number of rows that a query can return to the job manager by specifying an integer value on the **Maximum database results** setting.
 - c. Optionally change the Web address of the job manager that the administrative agent uses to retrieve jobs by specifying a Web address for the **Job manager URL** setting.
The Web address is used only when the job manager is configured as a proxy server.
 - d. Optionally specify an e-mail address on the E-mail sender's address setting.
The e-mail address that you specify is the e-mail address of the sender of the notification message that the job manager provides when jobs have completed.
 - e. Optionally select the **Start components as needed** setting to start components dynamically as needed for applications.
 - f. Optionally stop the job manager by clicking **Stop**.
Selecting this option stops the job manager and its console.
3. If you changed any of the settings, click **Apply**, and then **OK**.

Results

You configured the job manager with options that you selected.

What to do next

You can do other job management tasks such as administer node groups, view nodes, submit jobs, and view node resources.

Job manager settings

This panel allows you to configure the job manager server and view its properties. You can specify the default job expiration, the job manager Web address, and the mail provider Java Naming and Directory Interface (JNDI) name.

To view this administrative console page, click **System administration > Job manager**.

Name: Specifies the job manager server name. The name is read-only.

Default job expiration:

Specifies the default job expiration time in days.

Data type	Integer
Default	60

Maximum database results: The maximum database results property applies to find operations for jobs, nodes, and node resources. The number represents the maximum number of records that can be retrieved during a job manager find operation. This number can be further reduced by the maximum results property on a find operation. Assume, for example, that you specify the maximum results to display for finding nodes at 50, but the maximum database results property is set to 10000. If you have 20000 jobs, the find operation finds 50 nodes.

Data type	Integer
Default	10000

Job manager URL:

Specifies the Web address of the job manager that the administrative agent uses to fetch jobs.

The Web address that you specify is used only when the job manager is configured as a proxy server. The Web address overrides the default Web address. If you modify the Web address, you must reregister the nodes with the job manager. The change affects only the nodes previously registered.

Data type	String
Default	<code>http://host:port/otis/OMADMServlet</code>

The *host* and *port* are those of the job manager unless you use a Web server. In that case, change the *host* and *port* to that of the Web server.

Mail provider JNDI name:

Specifies an optional JNDI mail provider to send an e-mail notification that a job has completed.

Data type	String
------------------	--------

Default

None

E-mail sender's address: Specifies the e-mail address of the sender of the notification message that the job manager provides when jobs have completed. This setting is required if you specify a JNDI mail provider on the Mail provider JNDI name setting.

Start components as needed: Select this property if you want the server components started as they are needed for applications that run on this server.

When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the this product, make sure that those other products support this functionality before you select this property.

Process ID: Specifies the read-only process ID of the job manager.

Cell name: Specifies the read-only cell name of the job manager.

Node name: Specifies the read-only node name of the job manager.

State: Specifies the read-only state of the job manager, such as started or stopped.

Administration service settings

Use this page to view and change the configuration for an administration service.

To view this administrative console page, click **Servers > Application Servers > server_name > Administration > Administration Services**

Remote connector

Specifies the remote JMX Connector type. The remote JMX connector is the connector that is used between server processes that reside on different physical machines, for example, between the deployment manager and the node agent. Available options of SOAPConnector, RMIConnector, and JSR160RMI Connector are defined using the JMX Connectors page.

Data type

String

Default

SOAPConnector

Local connector

Specifies the local JMX Connector type. The local JMX connector is the connector used between server processes that reside on the same physical machine, for example, between the node agent and its application servers. Available options of SOAPConnector, RMIConnector, JSR160RMI Connector, and IPC Connector are defined using the JMX Connectors page.

Data type

String

Default

IPC Connector

Administration services custom properties

This topic discusses the administration services custom properties that you can set on the administrative console.

To view the administration services custom properties administrative console page that goes with this topic, click: **Servers > Application Server > *server_name* > Administration > Administration Services > Custom Property.**

Specify a property and its value as a name-value pair on the Administration services custom properties page.

com.ibm.websphere.mbeans.disableRouting

When a custom managed bean (MBean) is registered directly with the MBean server that runs in a WebSphere Application Server process, the MBean object name is enhanced by default to include the cell, node, and process names as key properties. To turn off the default behavior, set the following custom property on the application server.

With this enhancement, in a Network Deployment environment, the MBean that is registered on an application server is addressable through a client that is connected to the deployment manager.

If this custom property is set, an administrative client needs to connect directly to the application server on which the MBean is registered to invoke methods. The MBean cannot participate in all the distributed functions of the administrative system.

One or more MBean object names tagged with `<on>...</on>`. You can specify the object name of your MBean or a pattern that matches the names of several MBeans.

Example:

If you register a custom MBean with the `WebSphere:type=custom,name=custommbean1` object name and another custom MBean with the `WebSphere:type=custom,name=custommbean2` object name, each of the following values is valid:

- `<on>WebSphere:type=custom,name=custommbean1</on>`
The value disables the MBean object name modification for this MBean.
- `<on>WebSphere:type=custom,*</on>`
The value disables the MBean object name modification for both MBeans.
- `<on>WebSphere:type=custom,name=custommbean1</on><on>WebSphere:type=custom,name=custommbean2</on>`
The value disables the object name modification for both MBeans.

Administrative audits

This topic discusses aspects of administrative audits, such as log files that contain the audit information, the administrative actions that are audited, and the types of audit messages that are logged.

Administrative audits use the same logging facility as the rest of the product. The audits are available in both the `activity.log` file and the `SystemOut.log` of the server that performs the action. You do not need to enable trace to produce the audits. However, through the Repository service console page, you can control whether configuration change auditing is done. This type of audit is done by default. Operational command auditing is always enabled. Information about which user performed the change is available only when security is enabled.

You can do administrative audits with or without the security audit facility. The security audit facility can record unauthorized access in audit log files. You can sign and encrypt the file-based audit logs to ensure data integrity. You can protect the audit files using directory and file permissions.

The following administrative actions are audited:

- All configuration changes, in terms of the configuration documents that are created, modified, or deleted.
- Certain operational changes like starting and stopping nodes, clusters, servers, and applications. These managed bean (MBean) operations provide administrative auditing:

Table 1.

MBean type	MBean operations
CellSync	syncNode
Cluster	start, stop, stopImmediate, rippleStart
NodeAgent	launchProcess, stopNode, restart
Server	stop, stopImmediate
AppManagement	startApplication, stopApplication

Configuration change audits have ADMRxxxxI message IDs, where xxxx is the message number. Operational audits have ADMN10xxI message IDs, where 10xx is the message number.

Here are some audit examples from a Network Deployment environment.

The following audit example is from the deployment manager SystemOut.log file:

```
[7/23/03 17:04:49:089 CDT] 39c26dad FileRepositor A ADMR0015I: Document
cells/ellingtonNetwork/security.xml was modified by user u1.
[7/23/03 17:04:49:269 CDT] 3ea0edb5 FileRepositor A ADMR0016I: Document
cells/ellingtonNetwork/nodes/ellington/app.policy was created by user u1.
...
[7/23/03 17:13:54:081 CDT] 39a572a1 AdminHelper A ADMN1008I: Attempt
made to start the SamplesGallery application. (User ID = u1)
...
```

The following audit example is from the node agent SystemOut.log file:

```
[7/23/03 17:38:43:461 CDT] 23d1326 AdminHelper A ADMN1000I: Attempt
made to launch server1 on node ellington. (User ID = u1)
```

The following audit example is from the application server SystemOut.log file:

```
[7/23/03 17:39:59:360 CDT] 24865373 AdminHelper A ADMN1020I: Attempt
made to stop the server1 server. (User ID = u1)
```

The message text is split for printing purposes.

Remote file services

Configuration documents describe the available application servers, their configurations, and their contents. Two file services manage configuration documents: the file transfer service and the file synchronization service.

The following information describes what the file services do:

File transfer service

The file transfer service enables the moving of files between the deployment manager and the nodes as well as between the wsadmin scripting process and either the deployment manager or the application server. It uses the HTTP protocol to transfer files. When you enable security in the WebSphere Application Server product, the file transfer service uses certificate-based mutual authentication. You can use the default key files in a test environment. Ensure that you change the default key file to secure your system.

The ports used for file transfer are the HTTP_Transport port, the HTTPS transport port, the administrative console port, and the administrative console secure port. For more information, see the Planning for TCP/IP port convention topic in the *Installing your application serving environment* PDF.

File synchronization service

The file synchronization service ensures that a file set on each node matches that on the deployment manager node. This service promotes consistent configuration data across a cell. You can adjust several configuration settings to control file synchronization on individual nodes and throughout a system.

This service runs in the deployment manager and node agents, and ensures that configuration changes made to the cell repository are propagated to the appropriate node repositories. The cell repository is the master repository, and configuration changes made to node repositories are not propagated up to the cell. During a synchronization operation a node agent checks with the deployment manager to see if any configuration documents that apply to the node have been updated. New or updated documents are copied to the node repository, and deleted documents are removed from the node repository.

The default behavior, which is enabled, is for each node agent to periodically run a synchronization operation. You can configure the interval between operations or disable the periodic behavior. You can also configure the synchronization service to synchronize a node repository before starting a server on the node.

Configuring remote file services

Configuration data for the WebSphere Application Server product resides in files. Two services help you reconfigure and otherwise manage these files: the file transfer service and file synchronization service.

About this task

By default, the file transfer service is always configured and enabled at a node agent, so you do not need to take additional steps to configure this service. However, you might need to configure the file synchronization service.

1. Go to the File Synchronization Service page. Click **System Administration > Node agents** in the console navigation tree. Then, click the node agent for which you want to configure a synchronization server and click **File synchronization service**.
2. On the File Synchronization Service page, customize the service that helps make configuration data consistent across a cell by moving updated configuration files from the deployment manager to the node. Change the values for properties on the File Synchronization Service page. The file synchronization service is always started, but you can control how it runs by changing the values.
3. Optionally add a custom property for file synchronization.
 - a. Click **Custom properties** on the File Synchronization Service page.
 - b. Click **New**.
 - c. Specify a name and value for the custom property.

The `com.ibm.websphere.management.application.expand.wto` custom property:

Specify this custom property with a value of `true` if you want to display Enterprise Archive (EAR) file expansion errors on the z/OS operating system console. When the errors display, the operator can take appropriate corrective action for the failure.

Property	<code>com.ibm.websphere.management.application.expand.wto</code>
Data type	Boolean
Default	False

The `com.ibm.websphere.management.sync.allowfailure` custom property:

When synchronization fails five times consecutively, automatic synchronization is disabled. To keep automatic synchronization enabled, specify this custom property with a value of true.

Property	com.ibm.websphere.management.sync.allowfailure
Data type	Boolean
Default	False

File transfer service settings

Use this page to configure the service that transfers files from the deployment manager to individual remote nodes.

To view this administrative console page, click **System Administration > Node agents > *node_agent_name* > File transfer service**.

Enable service at server startup

Specifies whether the server attempts to start the specified service. Some services are always enabled and disregard this property if set. This setting is enabled by default.

Data type	Boolean
Default	true

Retries count

Specifies the number of times you want the file transfer service to retry sending or receiving a file after a communication failure occurs.

Data type	Integer
Default	3

If the retries count setting is blank, the file transfer service sets the default to 3. If the retries count setting is 0, the file transfer service does not retry. The default is the recommended value.

Retry wait time

Specifies the number of seconds that the file transfer service waits before it retries a failed file transfer.

Data type	Integer
Default	10

If the retry wait time setting is blank, the code sets the default to 10. If the retry wait time setting is 0, the file transfer service does not wait between retries. The default is the recommended value.

File synchronization service settings

Use this page to specify that a file set on one node matches that on the central deployment manager node and to ensure consistent configuration data across a cell.

You can synchronize files on individual nodes or throughout your system.

Note: If your installation includes mixed release cells, a large numbers of nodes, and runs a large number of applications, you might want to use the **Generic JVM Arguments** field, on the Java Virtual Machine Settings page of the administrative console, to enable the hot restart sync feature of the synchronization service. This feature indicates to the synchronization service that the installation is

running in an environment where configuration updates are not made when the deployment manager is not active. Therefore, the service does not have to perform a complete repository comparison when the deployment manager or node agent servers restart.

To view this administrative console page, click **System Administration > Node agents > *node_agent_name* > File synchronization service**.

Enable service at server startup

Specifies whether the server attempts to start the file synchronization service. This setting does not cause a file synchronization operation to start. This setting is enabled by default.

Data type	Boolean
Default	true

Synchronization Interval

Specifies the number of minutes that elapse between synchronizations. Increase the time interval to synchronize files less often. Decrease the time interval to synchronize files more often.

Data type	Integer
Units	Minutes
Default	10

The minimum value that the application server uses is 1. If you specify a value of 0, the application server ignores the value and uses the default of 1.

Automatic Synchronization

Specifies whether to synchronize files automatically after a designated interval. When this setting is enabled, the node agent automatically contacts the deployment manager every synchronization interval to attempt to synchronize the node's configuration repository with the master repository owned by the deployment manager.

If the Automatic synchronization setting is enabled, the node agent attempts file synchronization when it establishes contact with the deployment manager. The node agent waits the synchronization interval before it attempts the next synchronization.

For the z/OS platform, disablement of automatic synchronization is recommended when in a production environment or if you use the application rollout update capability.

Remove the check mark from the check box if you want to control when files are sent to the node.

Data type	Boolean
Default	true

Startup Synchronization

Specifies whether the node agent attempts to synchronize the node configuration with the latest configurations in the master repository prior to starting an application server.

The default is to not synchronize files prior to starting an application server. Enabling the setting ensures that the node agent has the latest configuration but increases the amount of time it takes to start the application server.

Note that this setting has no effect on the startServer command. The startServer command launches a server directly and does not use the node agent.

Data type	Boolean
------------------	---------

Default

false

Exclusions

Specifies files or patterns that should not be part of the synchronization of configuration data. Files in this list are not copied from the master configuration repository to the node, and are not deleted from the repository at the node.

The default is to have no files specified.

To specify a file, use a complete name or a name with a leading or trailing asterisk (*) for a wildcard. For example:

<code>cells/cell name/nodes/node name/file name</code>	Excludes this specific file
<code>*/file name</code>	Excludes files named <i>file name</i> in any context
<code>dirname/*</code>	Excludes the subtree under <i>dirname</i>

Press **Enter** at the end of each entry. Each file name appears on a separate line.

Since these strings represent logical document locations and not actual file paths, only forward slashes are needed no matter the platform.

Changes to the exclusion list are picked up when the node agent is restarted.

Data type

String

Units

File names or patterns

Stopping or canceling the z/OS location service daemon from the MVS console

Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP). This topic discusses how to issue MVS™ console commands to stop or cancel the z/OS location service daemon.

Before you begin

You must first install WebSphere Application Server.

About this task

If you cancel or stop the location service daemon, it cancels or stops all WebSphere Application Servers on the system. If you installed Network Deployment, the location service daemon also cancels or stops the deployment manager and the node agents.

Issue one of the following commands to stop the location service daemon:

```
STOP DAEMON01
CANCEL DAEMON01
CANCEL DAEMON01,ARMRESTART
```

If you issue the **stop** command, the server finishes all remaining work before shutting itself down. If you issue the **cancel** command, the server stops quickly. Inflight data and transactions might be lost. Use the **cancel** command that has the ARMRESTART option if automatic restart management (ARM) is active and you want the ARM to restart the location service daemon.

Results

You have stopped or cancelled the location service daemon.

Determining if the z/OS location service daemon is running

This article describes what steps you must follow to determine if the location service daemon is running on a z/OS system.

Before you begin

The location service daemon starts automatically if it is not already running when you start an Application Server or a deployment manager server.

About this task

Perform the following step any time you want to know whether the location service daemon is running.

To determine if the location service daemon is running, issue one of the display commands as described in the Example: Displaying active address spaces topic in the *Using the administrative clients* PDF , such as `d a,1`, from the MVS console.

Results

You know the system is running if you see the BBODMNx address space running, where x is a letter (A, B, C, and so on).

Modifying z/OS location service daemon settings

This article describes what steps you must follow to modify the location service daemon settings in an administrative console. Location service daemons provide the CORBA location service in support of Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP).

Before you begin

Complete the following items before starting this task:

- Configure the location service daemon

You must first configure the location service daemon in the WebSphere Application Server customization dialogs. After you configure the daemon, you can modify or view the location service daemon settings from the administrative console.

- Optionally enable security

There is no specific setting to enable Secure Sockets Layer (SSL) for the location service daemon. If you want to use the SSL protocol to encrypt communications to the location service daemon, complete the following items:

- Enable global security for the cell.
- Define a valid system SSL repertoire for the location service daemon.
- Set the z/OS user ID that is assigned to the location service daemon-started task to the same z/OS user ID that was used to create the key ring.

About this task

The location service daemon is an integral component of the Remote Method Invocation and Internet Inter-ORB Protocol (RMI/IIOP) communication function for the WebSphere Application Server for z/OS. The daemon works with z/OS workload management to distribute RMI requests (for example enterprise bean requests) among application servers in a cell.

When a client makes a remote call to an enterprise bean, a location service daemon determines which servers are eligible to process the request. The location service daemon makes the decision with the z/OS workload management function (WLM). The daemon then routes the request to the selected server, which establishes a CORBA session with the client. Subsequent calls to the same enterprise bean flow directly over the established session.

In a cell, one location service daemon exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group. An example of a system is the z/OS operating system on a logical partition (LPAR).

You can now modify the location service daemon settings in an administrative console.

1. Go to the Settings page for the location service daemon to view the help page.
2. For the administrative console, click **System Administration > Node groups > *sysplex node group* > z/OS location service daemon** in the console navigation tree. For the job manager administrative console, click **System Administration -> Job manager > z/OS Location Service** in the console navigation tree. For the administrative agent administrative console, click **System Administration -> Administrative agent > z/OS Location Service** in the console navigation tree.
3. Specify the following values for the location service daemon:
 - The job name
 - The ports on which it listens
 - The IP names on which it listens
 - The start command
 - The start command arguments
 - The SSL repertoire that it uses

Results

The location service daemon settings are modified.

z/OS location service daemon settings

In a cell, one location service daemon definition exists for each sysplex node group. A location service daemon process runs on each system that has a node in a sysplex node group in that cell. When a client makes a remote call to an enterprise bean, a location service daemon determines which server or servers are eligible to process the request, and routes the request to the selected server. An example of a system is the z/OS operating system on a logical partition (LPAR).

Changes made to these settings apply to the location service daemon in a sysplex node group.

This administrative console page is only visible if the node group contains z/OS nodes. To view this administrative console page, click **System Administration > Node groups > *sysplex node group* > z/OS location service daemon**.

Job name

Specifies the job name of the location service daemon. This name can be from one to eight characters. You can use alphanumeric or the national language characters of @#\$.

Data type String
Default None

Start command

Specifies the command string that servers use to automatically start the location service daemon.

Data type String
Format `START location service daemon JCL procedure name`

The procedure name is defined at the node level during customization. You can change the procedure name on the WebSphere Variables administrative console panels by going to **Environment > WebSphere variables**.

Example `START BBO6DMN`
Default `${NODE_DAEMON_START_COMMAND}`

`NODE_DAEMON_START_COMMAND` is the configuration variable name whose value you can change on the WebSphere Variables administrative console panels.

Listen IP name

Specifies the IP address on which the location service daemon listens. The Listen IP name must be either a dotted decimal IP address or an asterisk (*). The asterisk means that the location service daemon listens on all available IP addresses.

Data type String
Default asterisk (*)

Daemon IP name

Specifies the IP name that clients use to access enterprise beans and Common Object Request Broker Architecture (CORBA) components on servers that belong to the sysplex node group that this location service daemon serves.

The daemon IP name can be any of the following choices:

- IP name such as `www.foobar.com`
- IP address of the form `n.n.n.n`, where `n` is an integer
- Dynamic virtual IP address (DVIPA)

Data type String
Default None

Port

Specifies the port on which the location service daemon listens for Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 1.

Data type Integer
Default None
Range 1 to 65535

SSL port

Specifies the port on which the location service daemon listens for encrypted Remote Method Invocation and Internet Inter-ORB (RMI/IIOP) requests.

Because the system reserves port numbers less than 1024 for Transmission Control Protocol/Internet Protocol (TCP/IP) applications, the recommendation is to use port numbers greater than 1023. However, the port range starts at 0. A value of 0 disables the SSL port.

Data type	Integer
Default	None
Range	0 to 65535

SSL settings

Specifies a predefined list of Secure Sockets Layer settings for connections.

These settings are System SSL repertoires that you configured on the SSL repertoire panel. Select one repertoire from the list.

Changing the node host names

After creating a profile or adding a node, the host name of the server or its ports might be incorrect. You can follow the examples to change the server host name using command line tools and the wsadmin scripting tool, and the host name of the server ports using the administrative console and command line tools.

Before you begin

Create a profile or add a node to a cell. Verify that the host name of the server and the server ports are correct.

About this task

If the host name of a server or its ports is incorrect, then you might experience problems such as errors when you attempt to stop a server. One example task shows how to correct the server host name through command line tools and the wsadmin scripting tool. The other example task shows how to correct the host name of the server ports using the administrative console and command line tools.

- Correct the host name for an application server node, a node agent, or a deployment manager node using the wsadmin scripting tool and command line tools.

1. Launch the wsadmin tool.

Enter the following command:

```
wsadmin -lang jython
```

2. List the contents of the server configuration file.

Enter the following line of code:

```
AdminConfig.list('ServerIndex')
```

3. In the output, find the ServerIndex object for the application server node, the node agent, or the deployment manager, similar to the following examples:

Application server and node agent:

```
cells/isthmusCell116/nodes/isthmusNode06|serverindex.xml#ServerIndex_1
```

Deployment manager:

```
cells/isthmusCell116/nodes/isthmusCellManager06|serverindex.xml#ServerIndex_1
```

4. Modify the host name for the application server node, the node agent, or the deployment manager, similar to the following examples:

Application server and node agent:

Enter the following line of code:

```
AdminConfig.modify('(cells/isthmusCell116/nodes/isthmusNode06|serverindex.xml  
#ServerIndex_1)', "[[hostName new_host_name]]")
```

Deployment manager:

Enter the following line of code:

```
AdminConfig.modify('(cells/isthmusCell116/nodes/isthmusCellManager06|  
serverindex.xml#ServerIndex_1)', "[[hostName new_host_name]]")
```

The commands are split on multiple lines for printing purposes.

5. Verify that the host names are correct, similar to the following examples:

Application server and node agent:

Enter the following line of code:

```
AdminConfig.show('(cells/isthmusCell107/nodes/isthmusCellManager07|  
serverindex.xml#ServerIndex_1)', 'hostName')
```

The response is:

```
'[hostName isthmus]'
```

Deployment manager:

Enter the following line of code:

```
AdminConfig.show('(cells/isthmusCell107/nodes/isthmusNode04|  
serverindex.xml#ServerIndex_1)', 'hostName')
```

The response is:

```
'[hostName isthmus]'
```

The commands are split on multiple lines for printing purposes.

6. Save the configuration.

Enter the following line of code:

```
AdminConfig.save()
```

7. Type `exit` to end the `wsadmin` session.

8. If you changed the host names for the application server and node agent, update the node with the changes.

- a. Stop the node agent.

Enter the following command:

```
stopNode -profileName AppSrv01
```

- b. Stop the application server.

Enter the following command:

```
stopServer server1 -profileName AppSrv01
```

- c. Synchronize the nodes.

Enter the following command:

```
syncNode deployment_manager_host deployment_manager_port
```

- d. Restart the node agent.

Enter the following command:

```
startNode -profileName AppSrv01
```

- e. Restart the application server.

Enter the following command:

```
startServer server1 -profileName AppSrv01
```

9. If you changed the host name for the deployment manager, restart the deployment manager to apply the changes.

- a. Stop the deployment manager.
Enter the following command:
`stopManager -profileName DMgr01`
 - b. Start the deployment manager.
Enter the following command:
`startManager -profileName DMgr01`
- Correct the host names for the ports that an application server, node agent, or deployment manager opens.

If you have to correct the host names of the server ports, then you can make the correction using command line tools and either the wsadmin scripting tool or the administrative console. You might have to correct the host names of multiple ports for a particular server. This example shows you how to correct the host names using the administrative console and command line tools.

1. For the application server, select **Servers > Application servers > application server > Ports**. For the node agent, select **System administration > Node agents > node agent > Ports**. For the deployment manager, select **System administration > Deployment manager > Ports**.
2. Select a port whose host name needs changing.
3. Change the host name in the **Host** field; Click **OK**.
4. Continue selecting ports and changing host names until you correct each of the host names for the server ports.
5. Save the changes to the master configuration.
6. If you changed the host names for the application server and node agent, update the node with the changes.
 - a. Stop the node agent.
 - Select **System administration > Node agents**.
 - Select the node agent that you want to stop.
 - Click **Stop**.
 - b. Stop the application server.
 - Select **Servers > Application servers**.
 - Select the server that you want to stop.
 - Click **Stop**.
 - c. Synchronize the nodes.
Enter the following command:
`syncNode deployment_manager_host deployment_manager_port`
 - d. Restart the node agent.
Enter the following command:
`startNode -profileName AppSrv01`
 - e. Restart the application server.
 - Select **Servers > Application servers**.
 - Select the server that you want to restart.
 - Click **Start**.
7. If you changed the host name for the deployment manager, restart the deployment manager to apply the changes.
 - a. Stop the deployment manager.
 - Select **System administration > Deployment manager**.
 - Click **Stop**.
 - b. Start the deployment manager.
Enter the following command:

```
startManager -profileName DMgr01
```

Results

You have changed the host name of the server, the host names of the server ports, or both.

What to do next

You can continue to administer the product by doing such tasks as managing nodes, node agents, and node groups.

Administrative topology: Resources for learning

Use the following links to find relevant supplemental information about WebSphere Application Server administrative topologies and distributed administration. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the IBM WebSphere Application Server product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and IBM Redbooks® that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information:

Administration

- IBM WebSphere Application Server Redbooks

This site contains a listing of all WebSphere Application Server Redbooks.

- IBM WebSphere developerWorks®

This site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks zones, such as VisualAge® Java or WebSphere Application Server, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.

- WebSphere Application Server Support page

Take advantage of the Web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive Web-based support through the IBM Software Support portal at URL <http://www.ibm.com/software/support/> and search by product category, or by product name. For example, if you are experiencing problems specific to WebSphere Application Server, click **WebSphere Application Server** in the product list. The WebSphere Application Server Support page appears.

Extension MBean Providers collection

Use this page to view and change the configuration for JMX extension MBean providers.

You can configure JMX extension MBean providers to be used to extend the existing WebSphere managed resources in the core administrative system. Each MBean provider is a library containing an implementation of a JMX MBean and its MBean XML Descriptor file.

To view this administrative console page, click **Servers > Application Servers > *server_name* > Administration > Administration Services > Extension MBean Providers**

Name

The name used to identify the Extension MBean provider library.

Description

An arbitrary descriptive text for the Extension MBean Provider configuration.

Classpath

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path.

Extension MBean Provider settings

Use this page to view and change the configuration for a JMX extension MBean provider.

You can configure a library containing an implementation of a JMX MBean, and its MBean XML Descriptor file, to be used to extend the existing WebSphere managed resources in the core administrative system

To view this administrative console page, click **Servers > Application Servers > *server_name* > Administration > Administration Services > Extension MBean Providers > *provider_library_name***

Name

The name used to identify the Extension MBean provider library.

Data type String

Classpath

The path to the Java archive (JAR) file that contains the Extension MBean provider library. This class path is automatically added to the Application Server class path. The class loader needs this information to load and parse the Extension MBean XML Descriptor file.

Data type String

Description

An arbitrary descriptive text for the Extension MBean Provider configuration. Use this field for any text that helps identify or differentiate the provider configuration.

Data type String

Extension MBean collection

You can configure Java Management Extension (JMX) MBeans to extend the existing WebSphere Application Server managed resources in the administrative console. Use this page to register JMX MBeans. Any MBeans that are listed have already been registered.

To view this administrative console page, click **Servers > Application Servers > *server name* > Administration > Administration Services > Extension MBean Providers > *provider library name*> extensionMBeans**

DescriptorURI

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

Type Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Extension MBean settings

Use this page to view and configure Java Management Extension (JMX) MBeans.

To view this administrative console page, click **Servers > Application Servers > server name > Administration > Administration Services > Extension MBean Providers > provider library name > ExtensionMBeans > descriptorURI**

descriptorURI

Specifies the location, relative to the provider class path, where the MBean XML descriptor file is located.

Data type String

type

Specifies the type to use for registering this MBean. The type must match the type that is declared in the MBean descriptor file.

Data type String

Java Management Extensions connector properties

You can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, the scripts that run from a command-line interface, or a custom Java administrative client program that you write. You can also set SOAP connector properties in the soap.client.props file and IPC connector properties in the ipc.client.props file.

A Java Management Extensions (JMX) connector can be a Remote Method Invocation (RMI) connector, a Simple Object Access Protocol (SOAP) connector, a JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or an Inter-Process Communications (IPC) connector.

Note: You should eventually convert all of your RMI connectors to JSR160RMI connectors because support for the RMI connector is deprecated.

For specific information on how to code the JMX connector properties for the wsadmin tool, the Application Server commands, or scripts, see the particular tool or command. Read the application programming interfaces documentation to learn how to code the JMX connector properties for a custom Java administrative client program.

The JMX connectors that servers create use JMX connector properties that are accessible in the administrative console. The wsadmin tool and the Java administrative client use JMX connector properties in the soap.client.props, ipc.client.props, and sas.client.prop files.

For the administrative console, this topic specifies the coding of the particular setting or property. Coding of properties in the soap.client.props file and the ipc.client.props file that are specific to JMX connectors is specified. These SOAP properties begin with com.ibm.SOAP and the IPC properties begin with com.ibm.IPC. Other properties in the soap.client.props file and the ipc.client.props file that contain information that can be set elsewhere in the application server are not documented here. The coding for the com.ibm.ssl.contextProvider property, which can be set only in the soap.client.props file and the ipc.client.props file, is specified.

Each profile has property files at the following locations:

- For the SOAP connector:
 - *installation root/profiles/profile name/properties/soap.client.props*
- For the IPC connector:
 - *installation root/profiles/profile name/properties/ipc.client.props*

These property files allow you to set different properties, including security and timeout properties. These properties are the default for all the administrative connections that use either the SOAP JMX connector or the IPC JMX connector between processes that run in a particular profile. For instance, the wsadmin program running under a particular profile uses the property values from these files for the SOAP connector behavior and the IPC connector behavior unless the properties are overridden by some other programmatic means.

To view the JMX connector custom properties administrative console panel that goes with this article, click **Servers > Application servers > *server name* > Server Infrastructure > Administration > Administration Services > Additional properties > JMX Connectors > *connector type* > Additional Properties > Custom properties.**

SOAP connector properties

This section discusses JMX connector properties that pertain to SOAP connectors.

SOAP request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is 600 seconds. However, other components that connect to the SOAP client can override the default. Components that use the `soap.client.props` file have a default value of 180 seconds.

Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `soap.client.props` file.

Property	<code>com.ibm.SOAP.requestTimeout</code>
Data type	Integer
Range in seconds	0 to n
Default	If the property is zero (0), the request never times out. 180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>requestTimeout</code>
Data type	Integer
Range in seconds	0 to n
Default	If the property is zero (0), the request never times out. 600

- A Java administrative client. The property is `AdminClient.CONNECTOR_SOAP_REQUEST_TIMEOUT`.

Configuration URL

Specify the configuration Universal Resource Locator (URL) property if you want a program to read SOAP properties from this file. You can set the property by using one of the following options:

- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the application server on the `com.ibm.SOAP.ConfigURL` system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	ConfigURL
Data type	String
Valid Value	http://Path/soap.client.props
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_SOAP_CONFIG property.

Security context provider

This property indicates the Secure Sockets Layer (SSL) implementation to use between the application server and the SOAP client.

Set the property by using the soap.client.props file.

Property	com.ibm.ssl.contextProvider
Data type	String
Valid Values	IBMSSE2
Default	IBMSSE2

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between the application server and the SOAP client. Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The soap.client.props file.

Property	com.ibm.SOAP.securityEnabled
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	securityEnabled
Data type	Boolean
Default	False

- A Java administrative client. Use the AdminClient.CONNECTOR_SECURITY_ENABLED property.

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the soap.client.props file.

Property	com.ibm.ssl.alias
Data type	String
Default	DefaultSSLSettings

IPC connector properties

This section discusses JMX connector properties that pertain to IPC connectors.

IPC request timeout

The value that you choose depends on a number of factors, such as the size and the number of the applications that are installed on the server, the speed of your machine, and the usage of your machine.

The program default value for the request timeout is 600 seconds. However, other components that connect to the IPC client can override the default. Components that use the `ipc.client.props` file have a default value of 180 seconds.

Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	<code>com.ibm.IPC.requestTimeout</code>
Data type	Integer
Range in seconds	0 to n
Default	If the property is zero (0), the request never times out. 180

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>requestTimeout</code>
Data type	Integer
Range in seconds	0 to n
Default	If the property is zero (0), the request never times out. 600

- A Java administrative client. The property is `AdminClient.CONNECTOR_IPC_REQUEST_TIMEOUT`.

Configuration URL

Specify the configuration URL property if you want a program to read IPC properties from this file. You can set the property by using one of the following options:

- Scripts run from a command-line interface. Scripts can pass the Configuration URL property to the Application Server on the `com.ibm.IPC.ConfigURL` system property.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>ConfigURL</code>
Data type	String
Valid Value	<code>http://Path/ipc.client.props</code>
Default	None

- A Java administrative client. Use the `AdminClient.CONNECTOR_IPC_CONFIG` property.

Security context provider

This property indicates the SSL implementation to use between the application server and the IPC client.

Set the property by using the `ipc.client.props` file.

Property	<code>com.ibm.ssl.contextProvider</code>
Data type	String
Valid Values	IBMJSSE2
Default	IBMJSSE2

Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the IPC client. Set the property by using one of the following options:

- Scripts that run from a command-line interface.
- The `ipc.client.props` file.

Property	<code>com.ibm.IPC.securityEnabled</code>
Data type	Boolean
Default	False

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>securityEnabled</code>
Data type	Boolean
Default	False

- A Java administrative client. Use the `AdminClient.CONNECTOR_SECURITY_ENABLED` property.

SSL alias

This property specifies the alias to use for an SSL configuration for client connections. The value of the alias is what you want it to be.

Set the property in the `ipc.client.props` file.

Property	<code>com.ibm.ssl.alias</code>
Data type	String
Default	DefaultSSLSettings

SOAP, RMI, JSR160RMI, and IPC connector properties

This section discusses JMX connector properties that pertain to SOAP connectors, RMI connectors, JSR160RMI connectors, and IPC connectors.

Connector type

A connector type of SOAP, RMI, JSR160RMI, or IPC depends on whether the application server connects to a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The `wsadmin` tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	Type
-----------------	------

Data type	String		
Valid values	SOAPConnector RMIConnector JSR160RMIConnector IPCConnector		
Default	SOAPConnector	JSR160RMI	IPC

- A Java administrative client. Use the AdminClient.CONNECTOR_TYPE property. Specify the connector type by using the AdminClient.CONNECTOR_TYPE_RMI, the AdminClient.CONNECTOR_TYPE_SOAP, the AdminClient.CONNECTOR_TYPE_JSR160RMI, or the AdminClient.CONNECTOR_TYPE_IPC constants.

Host

The host name or the IP address of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts that run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	host
Data type	String
Valid values	Host name or IP address
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_HOST property.

Port

The port number of the server to which the application server connects. The server can be a SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	port
Data type	Integer
Valid value	Port number
Default	None

- A Java administrative client. Use the AdminClient.CONNECTOR_PORT property.

User name

The user name that the application server uses to access the SOAP server, an RMI server, a JSR160RMI server, or an IPC server. You can set the property by using one of the following options:

- The wsadmin tool.
- Scripts run from a command-line interface.

- The `soap.client.props` file for the SOAP server, an RMI server, a JSR160RMI server.

Property	<code>com.ibm.SOAP.loginUserId</code>
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	<code>com.ibm.IPC.loginUserId</code>
Data type	String
Valid value	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>username</code>
Data type	String
Valid value	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the `AdminClient.USERNAME` property.

Password

The password that the application server uses to access the SOAP server, the RMI server, the JSR160RMI server, or the IPC server. You can set the property by using one of the following options:

- The `wsadmin` tool.
- Scripts run from a command-line interface.
- The `soap.client.props` file for the SOAP server, the RMI server, or the JSR160RMI server.

Property	<code>com.ibm.SOAP.loginPassword</code>
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, or JSR160RMI.
Default	None

- The `ipc.client.props` file for the IPC server.

Property	<code>com.ibm.IPC.loginPassword</code>
Data type	String
Valid values	The value must match the global SSL settings for IPC.
Default	None

- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	<code>password</code>
Data type	String
Valid values	The value must match the global SSL settings for SOAP, RMI, JSR160RMI, or IPC.
Default	None

- A Java administrative client. Use the `AdminClient.PASSWORD` property.

Disabling a connector

You can enable or disable any of the JMX connectors from the administrative console.

- The wsadmin tool.
- The administrative console. Select the box next to the connector to enable the connector. Clear the box next to the connector to disable the connector.

Property	enabled
Data type	Boolean
Value	true false

RMI connector properties

This section discusses JMX connector properties that pertain to RMI connectors.

Disabling the JSR 160 RMI connector

Support for JMX Remote application programming interface (JSR 160) is enabled by default so that you automatically receive specification-compliant JMX function. To disable the function for a particular server, set the property by using one of the following options:

- The wsadmin tool.
- The administrative console. Specify the property and the value as a name-value pair on the JMX connector custom properties panel of the administrative console.

Property	disableJDKJMXConnector
Data type	string
Value	true

SOAP connector and Inter-Process Communications connector properties files

Use the soap.client.props file to set properties for the SOAP connector and the ipc.client.props file to set properties for the Inter-Process Communications (IPC) connector. Most of the properties in the ipc.client.props file have corresponding properties in the soap.client.props file.

The SOAP connector properties file for a particular profile is at the following location:

- *profile_root*/properties/soap.client.props

The IPC connector properties file for a particular profile is at the following location:

- *profile_root*/properties/ipc.client.props

The following table provides basic information on the various properties. Read the properties files to obtain more detailed information.

Some properties are split on multiple lines for printing purposes.

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP. securityEnabled	com.ibm.IPC. securityEnabled	Specifies enablement of security for the connector. Set the property to true to enable security.

SOAP connector properties	IPC connector properties	Description
com.ibm.SOAP.authenticationTarget	com.ibm.IPC.authenticationTarget	Specifies the type of authentication for the connector if security is enabled. You can specify <code><codeph>BasicAuth</codeph></code> for basic authentication. If no value is specified, basic authentication is used.
com.ibm.SOAP.loginUserId	com.ibm.IPC.loginUserId	Specifies the user ID for the connector if security is enabled, and you do not enter a user ID through a command prompt or standard in.
com.ibm.SOAP.loginPassword	com.ibm.IPC.loginPassword	Specifies the password for the connector if security is enabled, and you do not enter a password through a command prompt or standard in.
com.ibm.SOAP.loginSource	com.ibm.IPC.loginSource	Specifies automatic prompting for the user ID and password when you specify prompt. Prerequisites for using this property are discussed in the properties file for the particular connector.
com.ibm.SOAP.requestTimeout	com.ibm.IPC.requestTimeout	Specifies how long in seconds the connector waits for a server response. The property for the SOAP connector and the property for the IPC connector are each initially set to 180 in their respective properties files.
com.ibm.ssl.alias	com.ibm.ssl.alias	This property specifies the alias to use for a Secure Sockets Layer (SSL) configuration for client connections. The value of the alias is what you want it to be.
	timeToExpiration	Specifies the time in seconds that connections can be idle in the connection pool. Beyond this time the connections are purged. The initial setting for the property is 360.

Java Management Extensions connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors, which make connections between server processes.

To view this administrative console page, click one of the following paths:

- **Servers > Application Servers > *server_name* > Administration > Administration Services > JMX Connectors**
- **Servers > JMS Servers > *server_name* > Administration > Administration Services > JMX Connectors**

Java Management Extensions (JMX) connectors communicate with WebSphere Application Server when you invoke a scripting process. There is no default for the type and parameters of a connector. The `wsadmin.properties` file specifies the Simple Object Access Protocol (SOAP) connector and an appropriate

port number. You can also use the Remote Method Invocation (RMI) connector, the JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or the Inter-Process Communications (IPC) connector.

Use one of the following methods to select the connector type and attributes:

- Specify properties in a properties file.
- Indicate options on the command line.

On the z/OS platform, the SOAP connector, the RMI connector, the JSR160RMI connector, and the IPC connector connect to a controller (server) . WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type

Specifies the type of the JMX connector.

Data type

Enumeration

Default

SOAPConnector

Range

SOAPConnector

For JMX connections using Simple Object Access Protocol (SOAP).

RMIConnector

For JMX connections using Remote Method Invocation (RMI).

JSR160RMIConnector

For JMX connections using JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI).

IPCConnector

For JMX connections using Inter-Process Communications (IPC).

Enabled

Specifies whether a JMX connector is enabled. If Yes is specified, the connector is enabled. All JMX connectors are enabled by default.

Data type

Boolean

JMX connector settings

Use this page to view the configuration for a Java Management Extensions (JMX) connector, which makes connections between server processes.

To view this administrative console page, click one of the following paths:

- **Servers > Application Servers > *server_name* > Administration > Administration Services > JMX Connectors > *connector_type***

The Simple Object Access Protocol (SOAP) connector, the Remote Method Invocation (RMI) connector, the JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI) connector, or the Inter-Process Communications (IPC) connector connect to a controller (server). WebSphere Application Server for z/OS internally routes MBean requests from a controller to its servant regions as appropriate.

Type

Specifies the type of the JMX connector.

Data type	Enumeration
Default	SOAPConnector
Range	<p>SOAPConnector For JMX connections using Simple Object Access Protocol (SOAP).</p> <p>RMIConnector For JMX connections using Remote Method Invocation (RMI).</p> <p>JSR160RMIConnector For JMX connections using JMX Remote application programming interface (JSR 160) Remote Method Invocation (JSR160RMI).</p> <p>IPCCconnector For JMX connections using Inter-Process Communications (IPC).</p>

Repository service settings

Use this page to view and change the configuration for an administrative service repository.

To view this administrative console page, click **Servers > Application Servers > *server_name* Administration > Administration Services > Repository Service**.

Audit Enabled

Specifies whether to audit repository updates in the log file. The default is to audit repository updates.

Data type	Boolean
Default	true

Chapter 3. Working with server configuration files

This topic shows how to manage application server configuration files.

About this task

Application server configuration files define the available application servers, their configurations, and their contents.

A configuration repository stores configuration data.

By default, configuration repositories reside in the *config* subdirectory of the profile root directory.

A cell-level repository stores configuration data for the entire cell and is managed by a file repository service that runs in the deployment manager. The deployment manager and each node have their own repositories. A node-level repository stores configuration data that is needed by processes on that node and is accessed by the node agent and application servers on that node.

When you change a WebSphere Application Server configuration by creating an application server, installing an application, changing a variable definition or the like, and then save the changes, the cell-level repository is updated. The file synchronization service distributes the changes to the appropriate nodes.

You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

- Edit configuration files.

The master repository is comprised of .xml configuration files

You can edit configuration files using

- The administrative console. See the Using the administrative console topic in the *Using the administrative clients* PDF.
- Scripting. See the Getting started with scripting topic in the *Using the administrative clients* PDF.
- The wsadmin commands. See the Using command line tools topic in the *Using the administrative clients* PDF.
- Programming. See the Using administrative programs (JMX) topic in the *Using the administrative clients* PDF.
- By editing a configuration file directly.

The configuration files are in ASCII format. You cannot edit them in the Hierarchical File System (HFS). Instead, transfer the files to your workstation, edit them, and then transfer them back to the HFS.

- Save changes made to configuration files. Using the console, you can save changes as follows:
 1. In the navigation select **System Administration > Save changes to master repository**.
 2. Put a check mark in the **Synchronize changes with Nodes** check box.
 3. Click **Save**.
- Handle temporary configuration files resulting from a session timing out.
- Change the location of temporary configuration files.
- Change the location of backed-up configuration files.
- Change the location of temporary workspace files.
- Back up and restore configurations.

Configuration documents

WebSphere Application Server stores configuration data in several documents in a cascading hierarchy of directories. Most configuration documents have XML content.

The configuration documents describe the available application servers, their configurations, and their contents.

- “Hierarchy of directories of documents”
- “Changing configuration documents” on page 85
- “Transformation of configuration files” on page 85

Hierarchy of directories of documents

The cascading hierarchy of directories and the documents’ structure support multinode replication to synchronize the activities of all servers in a cell. In a Network Deployment environment, changes made to configuration documents in the cell repository, are automatically replicated to the same configuration documents that are stored on nodes throughout the cell.

At the top of the hierarchy is the **cells** directory. It holds a subdirectory for each cell. The names of the cell subdirectories match the names of the cells. For example, a cell named *cell1* has its configuration documents in the subdirectory *cell1*. The name of the cell must be different from the cluster name pair.

On the Network Deployment node, the subdirectories under the cell contain the entire set of documents for every node and server throughout the cell. On other nodes, the set of documents is limited to what applies to that specific node. If a configuration document only applies to *node1*, then that document exists in the configuration on *node1* and in the Network Deployment configuration, but not on any other node in the cell.

Each cell subdirectory has the following files and subdirectories:

- The *cell.xml* file, which provides configuration data for the cell
- Files such as *security.xml*, *virtualhosts.xml*, *resources.xml*, and *variables.xml*, which provide configuration data that applies across every node in the cell
- The **clusters** subdirectory, which holds a subdirectory for each cluster defined in the cell. The names of the subdirectories under clusters match the names of the clusters.

Each cluster subdirectory holds a *cluster.xml* file, which provides configuration data specifically for that cluster.

- The **nodes** subdirectory, which holds a subdirectory for each node in the cell. The names of the nodes subdirectories match the names of the nodes.

Each node subdirectory holds files such as *variables.xml* and *resources.xml*, which provide configuration data that applies across the node. Note that these files have the same name as those in the containing cell’s directory. The configurations specified in these node documents override the configurations specified in cell documents having the same name. For example, if a particular variable is in both cell- and node-level *variables.xml* files, all servers on the node use the variable definition in the node document and ignore the definition in the cell document.

Each node subdirectory holds a subdirectory for each server defined on the node. The names of the subdirectories match the names of the servers. Each server subdirectory holds a *server.xml* file, which provides configuration data specific to that server. Server subdirectories might hold files such as *security.xml*, *resources.xml* and *variables.xml*, which provide configuration data that applies only to the server. The configurations specified in these server documents override the configurations specified in containing cell and node documents having the same name.

- The **applications** subdirectory, which holds a subdirectory for each application deployed in the cell. The names of the applications subdirectories match the names of the deployed applications.

Each deployed application subdirectory holds a *deployment.xml* file that contains configuration data on the application deployment. Each subdirectory also holds a **META-INF** subdirectory that holds a Java 2

Platform, Enterprise Edition (J2EE) application deployment descriptor file as well as IBM deployment extensions files and bindings files. Deployed application subdirectories also hold subdirectories for all .war and entity bean .jar files in the application. Binary files such as .jar files are also part of the configuration structure.

An example file structure is as follows:

```
cells
  cell1
    cell.xml resources.xml virtualhosts.xml variables.xml security.xml
    nodes
      nodeX
        node.xml variables.xml resources.xml serverindex.xml
        serverA
          server.xml variables.xml
        nodeAgent
          server.xml variables.xml
      nodeY
        node.xml variables.xml resources.xml serverindex.xml
    applications
      sampleApp1
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
      sampleApp2
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
cells
  cell1
    cell.xml resources.xml virtualhosts.xml variables.xml security.xml
    nodes
      nodeX
        node.xml variables.xml resources.xml serverindex.xml
        serverA
          server.xml variables.xml
    applications
      sampleApp1
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
      sampleApp2
        deployment.xml
        META-INF
          application.xml ibm-application-ext.xml ibm-application-bnd.xml
```

Changing configuration documents

You can use one of the administrative tools (console, wsadmin, Java APIs) to modify configuration documents or edit them directly. It is preferable to use the administrative console because it validates changes made to configurations. “Configuration document descriptions” on page 86 states whether you can edit a document using the administrative tools or must edit it directly.

Transformation of configuration files

The WebSphere Application Server master configuration repository stores configuration files for all the nodes in the cell. When you upgrade the deployment manager from one release of WebSphere Application Server to another, the configuration files that are stored in the master repository for the nodes on the old release are converted into the format of the new release.

With this conversion, the deployment manager can process the configuration files uniformly. However, nodes on an old release cannot readily use configuration files that are in the format of the new release. WebSphere Application Server addresses the problem when it synchronizes the configuration files from the

master repository to a node on an old release. The configuration files are first transformed into the old release format before they ship to the node. WebSphere Application Server performs the following transformations on configuration documents:

- Changes the XML name space from the format of the new release to the format of the old release
- Strips out attributes of cell-level documents that are applicable to the new release only
- Strips out new resource definitions that are not understood by old release nodes

Configuration document descriptions

Most configuration documents have XML content. The table describes the documents and states whether you can edit them using an administrative tool or must edit them directly.

If possible, edit a configuration document using the administrative console because it validates any changes that you make to configurations. You can also use one of the other administrative tools (wsadmin or Java APIs) to modify configuration documents. Using the administrative console or wsadmin scripting to update configurations is less error prone and likely quicker and easier than other methods.

However, you cannot edit some files using the administrative tools. Configuration files that you must edit manually have an X in the **Manual editing required** column in the table below.

Document descriptions

(The paths in the Locations column are split on multiple lines for publishing purposes.)

Configuration file	Locations	Purpose	Manual editing required
admin-authz.xml	config/cells/ <i>cell_name/</i>	Define a role for administrative operation authorization.	
app.policy	config/cells/ <i>cell_name/</i> nodes/ <i>node_name/</i>	Define security permissions for application code.	X
cell.xml	config/cells/ <i>cell_name/</i>	Identify a cell.	
deployment.xml	config/cells/ <i>cell_name/</i> applications/ <i>application_name/</i>	Configure application deployment settings such as target servers and application-specific server configuration.	
filter.policy	config/cells/ <i>cell_name/</i>	Specify security permissions to be filtered out of other policy files.	X
integral-jms-authorizations.xml	config/cells/ <i>cell_name/</i>	Provide security configuration data for the integrated messaging system.	X
library.policy	config/cells/ <i>cell_name/</i> nodes/ <i>node_name/</i>	Define security permissions for shared library code.	X
namestore.xml	config/cells/ <i>cell_name/</i>	Provide persistent name binding data.	X
naming-authz.xml	config/cells/ <i>cell_name/</i>	Define roles for a naming operation authorization.	X

node.xml	config/cells/ cell_name/ nodes/node_name/	Identify a node.	
resources.xml	config/cells/ cell_name/ config/cells/ cell_name/ nodes/node_name/ config/cells/ cell_name/ nodes/node_name/ servers/ server_name/	Define operating environment resources, including JDBC, JMS, JavaMail, URL, JCA resource providers and factories.	
security.xml	config/cells/ cell_name/	Configure security, including all user ID and password data.	
server.xml	config/cells/ cell_name/ nodes/ node_name/ servers/ server_name/	Identify a server and its components.	
serverindex.xml	config/cells/ cell_name/ nodes/ node_name/	Specify communication ports used on a specific node.	
spi.policy	config/cells/ cell_name/ nodes/ node_name/	Define security permissions for service provider libraries such as resource providers.	X
variables.xml	config/cells/ cell_name/ config/cells/ cell_name/ nodes/ node_name/ config/cells/ cell_name/ nodes/node_name/ servers/ server_name/	Configure variables used to parameterize any part of the configuration settings.	
virtualhosts.xml	config/cells/ cell_name/	Configure a virtual host and its MIME types.	

Object names: What the name string cannot contain

When you create a new object using the administrative console or a wsadmin command, you often must specify a string for a name attribute.

Most characters are allowed in the name string. However, the name string cannot contain the following characters. The name string also cannot contain leading and trailing spaces.

/

forward slash

\	backslash
*	asterisk
,	comma
:	colon
;	semi-colon
=	equal sign
+	plus sign
?	question mark
	vertical bar
<	left angle bracket
>	right angle bracket
&	ampersand (and sign)
%	percent sign
'	single quote mark
"	double quote mark
]]>	No specific name exists for this character combination.
.	period (not valid if first character; valid if a later character)
#	Hash mark
\$	Dollar sign
~	Tilde

Handling temporary configuration files resulting from session timeout

If the console is not used for 15 minutes or more, the session times out. The same thing happens if you close the browser window without saving the configuration file. Changes to the file are saved to a temporary file when the session times out, after 15 minutes. This topic discusses what happens depending on whether you load the saved file.

Before you begin

A configuration file must have been saved from a previous administrative console session for the user ID that you are currently using to access the administrative console.

About this task

When a session times out, the configuration file in use is saved under the `userid/timeout` directory under the ServletContext's temp area. This value is the value of the `javax.servlet.context.tempdir` attribute of the ServletContext context. By default, it is: `profile_root/temp/hostname/Administration/admin/admin.war`

You can change the temp area by specifying it as a value for the `tempDir` init-param of the action servlet in the deployment descriptor (`web.xml`) of the administrative application.

The configuration file is also saved automatically when the same user ID logs into the non-secured console again, effectively starting a different session. This process is equivalent to forcing the existing user ID out of session, similar to a session timing out.

The next time you log on to the administrative console, you are prompted to load the saved configuration file. Do one of the following actions:

- Load the saved file.
 1. If a file with the same name exists in the `profile_root/config` directory, that file is moved to the `userid/backup` directory in the temp area.
 2. The saved file is moved to the `profile_root/config` directory.
 3. The file is then loaded.
- Do not load the saved file.

The saved file is deleted from the `userid/timeout` directory in the temp area.

Results

You loaded the saved configuration file if you chose to do so.

What to do next

Once you have logged into the administrative console, do whatever administration of WebSphere Application Server that you need to do.

Changing the location of temporary configuration files

You can change the default directory where temporary configuration files are stored.

About this task

The configuration repository uses copies of configuration files and temporary files while processing repository requests. It also uses a backup directory while managing the configuration. You can change the default locations of these files from the configuration directory to a directory of your choice by using the administrative console.

The default location for the configuration temporary directory is `profile_root/config/temp`. Use the administrative console to change the location of the temporary repository file location for all types of server processes. For example, to change the setting for Application Server, do the following steps:

1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click **server name > Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the temporary file location. The key for this property is `was.repository.temp`. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of backed-up configuration files

You can change the default directory where backup files are stored.

About this task

During administrative processes like adding a node to a cell or updating a file, configuration files are temporarily backed up to a backup location.

The default location for the backup configuration directory is `profile_root/config/backup`. Use the administrative console to change the location of the repository backup directory for all types of server processes. For example, to change the setting for Application Server, do the following steps:

1. Click **Servers > Application servers** in the navigation tree of the administrative console. Then, click **server name > Administration > Administration services > Repository service > Custom properties**.
2. On the Properties page, click **New**.
3. On the settings page for a property, define a property for the backup file location. The key for this property is `was.repository.backup`. The value is the full path name to the desired location.
4. Click **OK**.

Changing the location of the wstemp temporary workspace directory

Configuration changes are stored in the wstemp temporary workspace directory until the changes are merged with the master configuration repository. This topic discusses how to change the location of the wstemp temporary workspace directory.

Before you begin

You must first install WebSphere Application Server before you change the location of the wstemp directory, which is a temporary workspace directory.

About this task

Whenever a user logs into the administrative console, or uses wsadmin scripting to make a configuration change, the changes are stored in the workspace. When a user uses the ConfigService configuration service interface of the Java application programming interfaces (APIs), the user specifies a session object that is associated with the workspace in order to store the changes. Only when the user performs a save operation under the administrative console, wsadmin scripting, or the Java APIs are the changes propagated and merged with the master configuration repository. For each administrative console user or each invocation of wsadmin scripting, the application server creates a separate workspace directory to store the intermediate changes until the changes are merged with the master configuration repository. Users of the Java APIs use different session objects to decide where the workspace directory resides. Both the administrative console and wsadmin scripting generate user IDs randomly. The user IDs are different from the user IDs that you use to log into the administrative console or wsadmin scripting. The Java APIs can either randomly generate the user ID or specify the user ID as an option when creating the session object.

You might want to change the location of the wstemp directory if you want to keep it in a separate place from the product installation.

The product determines the location of the workspace in the following order by using the first Java Virtual Machine (JVM) property in the list that is set. If no JVM property is set, the product uses the default workspace location.

JVM System Property	Location	Comments
websphere.workspace.root	<p>The wstemp directory location is the value of the websphere.workspace.root JVM system property plus</p> <ul style="list-style-type: none">• /wstemp <p>For example, the websphere.workspace.root JVM system property and its value could be</p> <ul style="list-style-type: none">• -Dwebsphere.workspace.root=/temp <p>The property and its value are split on multiple lines for printing purposes.</p>	<p>Set the JVM system property for the deployment manager to change the wstemp directory location. Use the full path rather than a relative path for this property.</p>
If the websphere.workspace.root property is not set, the value of the user.install.root property is used.	<p>The default wstemp location is the value of the user.install.root JVM system property plus</p> <ul style="list-style-type: none">• /wstemp	<p>Do not change the user.install.root property as the profile creation process sets this property by pointing to the <i>profile_root</i> directory. In this case, the wstemp location is:</p> <ul style="list-style-type: none">• <i>profile_root</i>/wstemp

- Change the workspace location for a particular JVM property by setting the `-D` option on the **java** command.

This method of changing the workspace location is only needed when you run a standalone administrative program in local mode.

For example, use the following option:

```
-Dwebsphere.workspace.root=the location of the new workspace directory
```

- Change the JVM custom property through the administrative console by setting the JVM property as a name-value pair on the Custom properties page.

For example,

1. Click **Servers > Application Servers > *server_name* > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**.
2. Click **New**.
3. Specify `websphere.workspace.root` as the name.
4. Specify the full path of the new workspace directory as the value. The `wstemp` directory is created under that path.
5. Stop the server.
This step is optional if you want to keep your existing workspace files.
6. Copy files from the old location of the workspace directory to the new location of the workspace directory.
This step is optional if you want to keep your existing workspace files.
7. Start the server.
This step is optional if you want to keep your existing workspace files.

Results

You have used either the administrative console or the `-D` option on the **java** command to change the location of the `wstemp` temporary workspace directory.

Backing up and restoring administrative configuration files

This topic discusses how to back up and restore administrative configuration files.

About this task

WebSphere Application Server represents its administrative configurations as XML files. You should back up configuration files on a regular basis.

Restore the configuration only if the configuration files that you backed up are at the same level of the release, including fixes, as the release to which you are restoring.

1. Synchronize administrative configuration files.
 - a. Click **System Administration > Nodes** in the console navigation tree to access the Nodes page.
 - b. Click **Full Resynchronize**. The resynchronize operation resolves conflicts among configuration files and can take several minutes to run.
2. Run the `backupConfig` command to back up configuration files. See the `backupConfig` command topic in the *Using the administrative clients* PDF for information.
3. Run the `restoreConfig` command to restore configuration files. See the `restoreConfig` command topic in the *Using the administrative clients* PDF for information. Specify backup files that do not contain invalid or inconsistent configurations.

Backing up the WebSphere Application Server for z/OS system

This topic discusses methods for backing up configuration and data for the WebSphere Application Server for z/OS system.

About this task

Use the following guidelines to back up parts of your WebSphere Application Server for z/OS system.

1. **Back up the HFS that contains your WebSphere Application Server for z/OS configuration** (e.g. `WebSphere/V5R0M0/AppServer`).
2. Back up the RMDATA log for Resource Recovery Service (RRS). Otherwise, a failure could force you to do a cold start of RRS.
3. Set the ARCHIVE log retention period to one day.
4. Incorporate the following items in your normal backup procedures:
 - WebSphere Application Server for z/OS procedure libraries.
 - WebSphere Application Server for z/OS load libraries.
 - The directory where WebSphere Application Server for z/OS run-time information is written. The default is `/WebSphere/V5R0M0`.
5. Back up your own application executable files, databases, and bindings.
6. If you wish to back up a single server, you can use the export/import function in the Administrative Console. For details on how to do this, see the assembling applications information.

Server configuration files: Resources for learning

Use the following links to find relevant supplemental information about administering WebSphere Application Server configuration files. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the IBM WebSphere Application Server product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and IBM Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information:

Administration

- IBM WebSphere Application Server Redbooks

This site contains a listing of all WebSphere Application Server Redbooks.

- IBM WebSphere developerWorks

This site is the home of technical information for developers working with WebSphere products. You can download WebSphere software, take a fast path to developerWorks zones, such as VisualAge Java or WebSphere Application Server, learn about WebSphere products through a newcomers page, tutorials, technology previews, training, and Redbooks, get answers to questions about WebSphere products, and join the WebSphere community, where you can keep up with the latest developments and technical papers.

- WebSphere Application Server Support page

Take advantage of the Web-based Support and Service resources from IBM to quickly find answers to your technical questions. You can easily access this extensive Web-based support through the IBM Software Support portal at URL <http://www.ibm.com/software/support/> and search by product category, or by product name. For example, if you are experiencing problems specific to WebSphere Application Server, click **WebSphere Application Server** in the product list. The WebSphere Application Server Support page appears.

Chapter 4. Administering application servers

An application server configuration provides settings that control how an application server provides services for running applications and their components.

About this task

After you install the product, you might have to perform one or more of the following tasks. Unless the task you want to perform is dependent on the existence of an application server, you can perform these tasks in any order.

- Create an application server.
- Create clusters for workload balancing.
- Configure the server startup process such that only server components that are initially needed are started.

When the server is configured such that only the components that are initially needed are started during the startup process, the remaining components are dynamically started as they are needed.

Note: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.

- Configure transport chains to handle client requests.
- Develop custom services.
- Define processes for the application server.

As part of defining processes, you might want to:

1. Define process execution statements for starting or initializing a UNIX® process.
2. Configure monitoring policies to track the performance of a process.
3. Configure the process logs to which standard out and standard error streams write.
4. Configure name-value pairs for properties.

- Configure the Java virtual machine.
- Optional: Run the `updateZOSStartArgs` script to modify the command that is used to start the application server.

Running this command enables z/OS to reuse the z/OS ASID that is associated with the controller.

Note: Before running this script, verify that you are running on z/OS Version 1.9 or higher, and that the `REUSASID(YES)` keyword is included in the z/OS `DIAGxx PARMLIB` member. This `REUSASID` option determines whether z/OS is enabled to reuse all ASIDs for the operating system image, including those that are associated with cross-process services. If the `REUSASID` option is set to `NO`, the ASIDs associated with the controller will not be reused even if you have run the `updateZOSStartArgs` script to enable this function for the product

Results

Any new application servers you create are displayed in the list of servers on the administrative console Application servers page.

What to do next

- Manage your application servers. Any newly created application servers are configured with many default settings that do not display when you run the Create New Application Server wizard. You might need to change some of these settings to better fit the needs of your environment.
- Deploy an application or component on the application server.
- View the status of the applications running on the application server.

Virtual hosts

A virtual host is a configuration entity that enables a single host machine to resemble multiple host machines. It maintains a list of Multipurpose Internet Mail Extensions (MIME) types that it processes. You can associate a virtual host to one or more Web modules, but you can associate each Web module with one and only one virtual host. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Each virtual host has a logical name and a list of one or more DNS aliases by which it is known. A DNS alias is the TCP/IP hostname and port number that is used to request the servlet, for example `yourHostName:80`. When no port number is specified, 80 is assumed.

The virtual host configuration uses wildcard entries with the ports for its virtual host entries.

- The default alias is `*:80`, using an external port that is not secure.
- Aliases of the form `*:9080` use the internal port that is not secure.
- Aliases of the form `*:9443` use the secure internal port.
- Aliases of the form `*:443` use the secure external port.

A client request for a servlet, JavaServer Pages file, or related resource contains a DNS alias and a Uniform Resource Indicator (URI) that is unique to that resource. When a client request for a servlet, JavaServer Pages file, or related resource is received, the DNS alias is compared to the list of all known virtual host groups to locate the correct virtual host, and the URI is compared to the list of all known URI groups to locate the correct URI group. If the virtual host group and URI group are found, the request is sent to the corresponding server group for processing and a response is returned to browser. If a matching virtual host group or URI group is not found, an error is returned to the browser.

A virtual host is not associated with a particular node (machine). It is a configuration, rather than a live object, which is why you can create it, but cannot start or stop it. A default virtual host, named `default_host`, is automatically configured the first time you start an application server. Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

The DNS aliases for the default virtual host are configured as `*:80` and `*:9080`, where port 80 is the HTTP server port and port 9080 is the port for the default server's HTTP transport. The default virtual host includes common aliases, such as the machine's IP address, short host name, and fully qualified host name. One of these aliases comprises the first part of the path for accessing a resource such as a servlet. For example, the alias `localhost:80` is used in the request `http://localhost:80/myServlet`.

Adding a `localhost` to the virtual hosts adds the host name and IP address of the `localhost` machine to the alias table. This allows a remote user to access the administrative console.

You can use the administrative console to add or change DNS aliases if you want to use ports other than the default ports. If you do make a change to a DNS alias, you must regenerate the Web server plug-in configuration. You can use the administrative console to initiate the plug-in regeneration.

Note: You might want to add additional aliases or change the default aliases if:

- The HTTP server instance is running on a port other than 80. Add the correct port number to each of the aliases. For example, change `yourhost` to `yourhost:8000`.
- You want to make HTTPS requests, which use Secure Sockets Layer (SSL). To make HTTPS requests you must add port 443 to each of the aliases. Port 443 is the default port for SSL requests.
- Your Web server instance is listening for SSL requests on a port other than 443. In this situation, you must add that port number to each of the aliases.
- You want to use a port other than default port (9080) for the application server.
- You want to use other aliases that are not listed.

When you request a resource, the product tries to map the request to an alias of a defined virtual host. The `http://host:port/` portion of the virtual host is not case sensitive, but the URL that follows is case sensitive. The match for the URL must be alphanumerically exact. Different port numbers are treated as different aliases.

For example, the request `http://www.myhost.com/myservlet` maps successfully to `http://WWW.MYHOST.COM/myservlet` but not to `http://WWW.MYHOST.COM/MYSERVLET` or `Www.Myhost.Com/Myservlet`. In the latter two cases, these mappings fail because of case sensitivity. The request `http://www.myhost.com/myservlet` does not map successfully to `http://myhost/myservlet` or to `http://myhost:9876/myservlet`. These mappings fail because they are not alphanumerically correct.

You can use wildcard entries for aliases by port and specify that all valid host name and address combinations on a particular port map to a particular virtual host.

If you request a resource using an alias that cannot be mapped to an alias of a defined virtual host, you receive a 404 error in the browser that you used to issue the request. A message states that the virtual host could not be found.

Two sets of associations occur for virtual hosts. Application deployment associates an application with a virtual host. Virtual host definitions associate the network address of the machine and the HTTP transport or Web server port assignment of the application server with the virtual host. Looking at the flow from the Web client request for the snoop servlet, for example, the following actions occur:

1. The Web client asks for the snoop servlet: at Web address `http://www.some_host.some_company.com:9080/snoop`
2. The `some_host` machine has the 9080 port assigned to the standalone application server, `server1`.
3. `server1` looks at the virtual host assignments to determine the virtual host that is assigned to the alias `some_host.some_company.com:9080`.
4. The application server finds that no explicit alias for that DNS string exists. However, a wild card assignment for host name `*` at port 9080 does exist. This is a match. The virtual host that defines the match is `default_host`.
5. The application server looks at the applications deployed on the `default_host` and finds the snoop servlet.
6. The application server serves the application to the Web client and the requester is able to use the snoop servlet.

You can have any number of aliases for a virtual host. You can even have overlapping aliases, such as:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
	my_machine	9080
	my_machine.my_company.com	9080
	localhost	80

The Application Server looks for a match using the explicit address specified on the Web client address. However, it might resolve the match to any other alias that matches the pattern before matching the explicit address. Simply defining an alias first in the list of aliases does not guarantee the search order whenever the product is looking for a matching alias.

A problem can occur if you use the same alias for two different virtual hosts. For example, assume that you installed the default application and the snoop servlet on the default_host. You also have another virtual host called the admin_host. However, you have not installed the default application or the snoop servlet on the admin_host.

Assume that you define overlapping aliases for both virtual hosts because you accidentally defined port 9080 for the admin_host instead of port 9060:

Virtual host	Alias	Port
default_host	*	9080
	localhost	9080
admin_host	*	9060
	my_machine.com	9080

Assume that a Web client request comes in for `http://my_machine.com:9080/snoop`.

If the application server matches the request against `*:9080`, the application is served from the default_host. If the application server matches the request to `my_machine.com:9080`, the application cannot be found. A 404 error occurs in the browser that issues the request. A message states that the virtual host could not be found.

This problem is the result of not finding the requested application in the first virtual host that has a matching alias. The correct way to code aliases is for the alias name on an incoming request to match only one virtual host in all of your virtual host definitions. If the URL can match more than one virtual host, you can see the problem just described.

Configuring virtual hosts

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers each on their own host machine. You can separate and control which resources are available for client requests by combining multiple host machines into a single virtual host, or by assigning host machines to different virtual hosts.

Before you begin

If your external HTTP server configuration uses the default port, 9080, you do not have to perform these steps.

About this task

Virtual hosts isolate and independently manage multiple sets of resources on the same physical machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host. This is true even though the virtual hosts share the same application server on the same physical machine.

For example, suppose that:

- An Internet service provider (ISP) has two customers with Internet sites hosted on the same machine. The ISP keeps the two sites isolated from one another, despite their sharing a machine, by using virtual hosts. The ISP associates the resources of the first company with VirtualHost1 and the resources of the second company with VirtualHost2. Both virtual hosts map to the same application server.
- Both company sites offer the same servlet. Each site has its own instance of the servlet, and is unaware of the same servlet on the other site. If the company whose site is organized on VirtualHost2

is past due in paying its account with the ISP, the ISP can refuse all servlet requests that are routed to VirtualHost2. Even though the same servlet is available on VirtualHost1, the requests directed at VirtualHost2 do not go to the other virtual host.

Because the servlet is associated with a virtual host instead of the actual DNS address, The servlet on virtual host VirtualHost1 does not share its context with the servlet that has the same name on virtual host VirtualHost2. Requests for the servlet on VirtualHost1 can continue as usual, even though VirtualHost2 is refusing to fill requests for the servlet with the same name.

If any of the following conditions exist, you must update the HTTP port numbers associated with the default virtual host. or define a new virtual host and associate it with the ports your HTTP server configuration uses:

- Your external HTTP server configuration uses a port other than the default port of 9080, you must define the port that you are using.
- You are using the default HTTP port 9080, but the port is no longer defined. You must define port 9080.
- You have created multiple application servers as either stand-alone servers or cluster members, and these servers use the same virtual host. Because each server must be listening on a different port, you must define a virtual host alias for the HTTP port of each server.

If you define new virtual host aliases, identify the port values that the aliases use on the Host alias settings page in the administrative console.

Perform the following steps to create a new virtual host or change the configuration of an existing virtual host.

1. In the administrative console, click **Environment > Virtual hosts**.
2. Optional: Create a new virtual host. If you create a new virtual host, a default set of 90 MIME entries are automatically created for that virtual host.
 - a. In the administrative console, click **New**.
 - b. Enter the name of the new virtual host and click **OK**. The new virtual host appears in the list of virtual hosts you can configure.
3. Select the virtual host whose configuration you want to change.
4. Under Additional Properties, click **Host aliases**.
5. Create new host aliases or update existing host aliases to associate each of your HTTP port numbers with this virtual host.

There must be a virtual host alias corresponding to each port your HTTP server configuration uses. There is one HTTP port associated with each Web container, and it is usually assigned to the virtual host named `default_host`. You can change the default assignment to any valid virtual host.

The host aliases associated with the `default_host` virtual host are set to `*` when you install the product. The `*` (an asterisk) indicates that the alias name does not have to be specified or that any name can be specified.

When the URL for the application is entered into a Web browser, the port number is included. For example, if 9082 is the port number, the specified URL might look like the following:

```
http://localhost:9082/wlm/SimpleServlet
```

To create a new host alias:

- a. Click **New**.
- b. Specify a host alias name in the Host Name field and one of your HTTP ports in the Port field.
You can specify `*` (an asterisk) for the alias name if you do not want to require the specification of the alias name or if you want to allow any name to be specified.
- c. Click **OK** and **Save** to save your configuration change.

To update an existing host alias:

- a. Select an existing host alias name.

- b. Change the value specified in the Port field to one of your HTTP ports.
 - c. Click **OK** and **Save** to save your configuration change.
6. Optional: Define a MIME object type and its file name extension if you require a MIME type other than the pre-defined types.
 - a. For each needed MIME entry on the MIME type collection page, click **New**.
 - b. On the MIME type settings page, specify a MIME type and extension.
 - c. Click **OK** and **Save** to save your configuration change.
7. Regenerate the Web server plug-in configuration.
 - a. **Servers > Server Types > Web servers**, then select the appropriate Web server.
 - b. Click **Generate pug-in**, then click **Propagate plug-in**.
8. Restart the application server.

Virtual host collection

Use this page to create and manage configurations that each let a single host machine resemble multiple host machines. Such configurations are known as *virtual hosts*.

To view this administrative console page, click **Environment > Virtual hosts**.

Each virtual host has a logical name (which you define on this panel) and is known by its list of one or more domain name system (DNS) aliases. A DNS alias is the TCP/IP host name and port number used to request the servlet, for example yourHostName:80. (Port 80 is the default.)

You define one or more alias associations by clicking an existing virtual host or by adding a new virtual host.

When a servlet request is made, the server name and port number entered into the browser are compared to a list of all known aliases in an effort to locate the correct virtual host to serve the servlet. No match returns an error to the browser.

An application server profile provides a default virtual host with some common aliases, such as the internet protocol (IP) address, the DNS short host name, and the DNS fully qualified host name. The alias comprises the first part of the path for accessing a resource such as a servlet.

For example, the alias is localhost:80 in the request `http://localhost:80/myServlet`.

A virtual host is not associated with a particular profile or node (machine), but is associated with a particular server instead. It is a configuration, rather than a "live object." You can create a virtual host, but you cannot start or stop it.

For many users, creating virtual hosts is unnecessary because the default_host that is provided is sufficient.

Adding the host name and IP address of the localhost machine to the alias table lets a remote user access the administrative console.

Resources associated with one virtual host cannot share data with resources associated with another virtual host, even if the virtual hosts share the same physical machine.

Name

Specifies a logical name for configuring Web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Virtual hosts enable you to isolate, and independently manage, multiple sets of resources on the same physical machine. Determine whether you need a virtual host alias for each port associated with an HTTP transport channel or an HTTP transport. There must be a virtual host alias corresponding to each port used by an HTTP transport channel or an HTTP transport. There is one HTTP transport channel or HTTP transport associated with each Web container, and there is one Web container in each application server.

When you create a virtual host, a default set of 90 MIME entries is created for the virtual host.

You must create a virtual host for each HTTP port in the following cases:

- You use the internal HTTP transport with a port other than the default value of 9080, or for some reason the virtual host does not contain the usual entry for port 9080.
- You create multiple application servers, such as stand-alone servers, managed servers, or cluster members, that are using the same virtual host. Because each server must be listening on a different HTTP port, you need a virtual host alias for the HTTP port of each server.

Virtual host settings

Use this page to configure a virtual host instance.

To view this administrative console page, click **Environment** > **WebSphere variables***virtual_host_name*.

Name:

Specifies a logical name for configuring Web applications to a particular host name. The default virtual host is suitable for most simple configurations.

Data type	String
Default	default_host

Host alias collection

Use this page to manage host name aliases defined for a virtual host. An alias is the DNS host name and port number that a client uses to form the URL request for a Web application resource.

To view this administrative console page, click **Environment** > **Virtual host***virtual_host_name* > **Host aliases**.

Host name:

Specifies the IP address, DNS host name with domain name suffix, or just the DNS host name, used by a client to request a Web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page). For example, the host alias name is *myhost* in a DNS name of *myhost:8080*.

The product provides a default virtual host (named *default_host*). The virtual host configuration uses the wildcard character * (asterisk) along with the port number for its virtual host entries. Unless you specifically want to isolate resources from one another on the same node (physical machine), you probably do not need any virtual hosts in addition to the default host.

Port:

Specifies the port for which the Web server has been configured to accept client requests. For example, the port assignment is *8080* in a DNS name of *myhost:8080*. A URL refers to this DNS as:
http://myhost:8080/servlet/snoop.

Host alias settings:

Use this page to view and configure a host alias.

To view this administrative console page, click **Environment > Virtual hosts > virtual_host_name > Host aliases > host_alias_name**.

Host name:

Specifies the IP address, domain name system (DNS) host name with domain name suffix, or the DNS host name that clients use to request a Web application resource, such as a servlet, JSP file, or HTML page.

For example, when the DNS name is myhost, the host alias is myhost:8080, where 8080 is the port. A URL request can refer to the snoop servlet on the host alias as: http://myhost:8080/servlet/snoop.

When there is no port number specified for a host alias, the default port is 80. For existing virtual hosts, the default host name and port reflect the values specified at product installation or configuration. For new virtual hosts, the default can be * to allow any value or no specification.

Data type	String
Default	*
	You can also use the IP address or the long or short DNS name.

Port:

Specifies the port where the Web server accepts client requests. Specify a port value in conjunction with the host name.

Specifies the port where the virtual host accepts Web client requests. The port number that you specify must be a unique in conjunction with the host name to avoid conflicts with other virtual hosts. The port number default is port 80, which is the default Web server port. You can assign another port number if you want to use the internal HTTP transport capability of the application server, or to use another port that you have designated as the Web server port. For example, you can create a new virtual host and assign port 9085 to that virtual host if you want to serve application resources over the internal HTTP transport of the application server that uses port 9085.

Data type	Integer
Default	80

MIME type collection

Use this page to view and configure multi-purpose internet mail extensions (MIME) object types and their file name extensions.

The list shows a collection of MIME type extension mappings defined for the virtual host. Virtual host MIME entries apply when you do not specify MIME entries at the Web module level.

To view a list of current virtual host Mime types in the administrative console, click **Environment > Virtual hosts virtual_host_name > Mime types**.

MIME type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is text/html.

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a text/html MIME type are htm and html.

MIME type settings:

Use this page to configure a multi-purpose internet mail extensions (MIME) object type.

To view this administrative console page, click **Environment > Virtual hosts***virtual_host_name* > **Mime types > mime_type**.

MIME type:

Specifies a MIME type, which can be application, audio, image, text, video, www, or x-world. An example value for MIME type is text/html.

An example value for MIME type is text/html. A default value appears only if you are viewing the configuration for an existing instance.

Data type String

Extensions:

Specifies file extensions of files that map the MIME type. Do not specify the period before the extension. Example extensions for a text/html MIME type are htm and html.

File extensions for a text/html MIME type are .htm and .html. A default value appears only if you are viewing the configuration for an existing MIME type.

Data type String

Creating, editing, and deleting WebSphere variables

You can use WebSphere variables to provide settings for any of the string data type attributes that are contained in the product configuration files.

Before you begin

Because applications cannot directly access WebSphere variables, if you define a WebSphere variable inside of an application, an error message, such as "Unknown variable," is returned. If you must reference a WebSphere variable from within an application, include the following method in the application to expand the string that uses the WebSphere variable:

```
private String expandVariable(String s) throws
javax.management.JMException {
    com.ibm.websphere.management.AdminService as =
    com.ibm.websphere.management.AdminServiceFactory.getAdminService
    ();

    String server = as.getProcessName();

    java.util.Set result = as.queryNames(new javax.management.ObjectName("*:*,type=AdminOperations,process="
    + server), null);

    return (String)as.invoke((javax.management.ObjectName)
    result.iterator().next(),"expandVariable",new Object[]
    {"${"+s+"}"}, new String[] {"java.lang.String"});
```


About this task

WebSphere variables are usually used to specify file paths. The "Variable settings" topic supplies further details about specifying variables and highlights further details about product components that use them.

WebSphere variables are also used to configure:

- Product path names, such as JAVA_HOME, and APP_INSTALL_ROOT.
- Configure certain cell-wide or cluster-wide customization values.
- The location service.
- Environment variables.

The variable scoping mechanism for WebSphere variables enables you to define a variable at the node, cluster, or cell level, as well as at the server level. This mechanism enables you to specify a setting for all of the servers in a node, cluster, or cell, instead of individually specifying the setting for each server.

To define a new variable, change the value of an existing variable, or delete an existing variable complete the following steps, as appropriate.

1. Click **Environment > WebSphere variables** in the administrative console
2. Select the scope of the variable from the list of available scopes.

If you create a new variable, it will be created at the selected scope. If you define the same variable at multiple levels, the more granular definition overrides the higher level setting. For example, if you specify the same variable on a cell level and at a node level, the node level setting overrides the cell level setting.

Scoping variables is particularly important if you are testing data source objects. Variable scoping can cause a data source to fail the test connection, but to succeed at run time, or to pass the test connection, but fail at run time.

3. Create a new variable.

- a. Click **New**.

- b. Specify a name, a value, and, optionally, a description for the variable.

The application server uses WebSphere Application Server internal variables for its own purposes. The prefixes that indicate that a variable is internal are WAS_DAEMON_<server custom property>, WAS_DAEMON_ONLY_<server custom property>, and WAS_SERVER_ONLY_<server custom property>. Any variables with these tags are not intended for your use. They are reserved exclusively for use by the server run time. Modifying these variables can cause unexpected errors.

You can use WebSphere variables to modify the daemon configuration. By appending a server custom property onto a daemon tag, you can designate that variable specifically for that daemon. Enter DAEMON_<server custom property> in the **Name** field. For example, if you enter DAEMON_ras_trace_outputlocation in the Name field and SYSOUT in the Value field, you can direct that particular daemon's trace output to SYSPRINT.

You can create WebSphere variables that support substitution. For example, if you enter \${<variable name>} in the **Name** field, the value of <variable name> becomes the name of your new WebSphere variable. For example if you enter \${JAVA_HOME} as the name of your variable, the name of the WebSphere variable that is created is the Java home directory.

- c. Click **OK**.

- d. Click **Environment > WebSphere variables** in the administrative console navigation, and verify that the variable is displayed in the list of variables for the selected scope.

The administrative console does not pick up typing errors. The variable is ignored if it is referred to incorrectly.

4. Modify the setting for an existing variable.

- a. Click on the name of the variable that you want to change.
- b. Modify the content of the Values field.

The Values field for some of the variables that are already defined when you install the product are read-only because changing the values that are specified for those variables might cause product processing errors.

- c. Click **OK**.
5. Delete an existing variable.
 - a. Select the variable that you want to delete.
 - b. Click **Delete**.
 - c. Click **OK**.
 - d. Verify that this variable was removed from the list of variables for the selected scope.
6. Save your configuration.
7. Stop the affected servers and start those servers again to put the variable configuration change into effect.

If the change you made affects a node, you must stop and restart all of the servers on that node. Similarly if the change you made affects a cell, you must stop and restart all of the servers in that cell.

WebSphere variables collection

Use this page to view and change the defined product variables with their values. You can also use this page to create a new variable, or delete an existing variable. These variables are name and value pairs that are used to provide the settings for the string data type configuration attributes that are contained in one of the XML formatted configuration files that reside in the product repository.

To view this administrative console page, click **Environment > WebSphere variables**.

To display a list of all of the variables that are defined for a specific scope, select that scope.

To view additional information about a specific variable, or to change the setting for that variable, click the variable name. Some of the pre-defined variables, that is, variables that already exist when you install the product, are set at values that are required for the product to function properly. The Value fields for these variables are read-only and cannot be edited.

To define a new variable, select the appropriate scope from the list of available options and then click **New**. The selected scope indicates the level at which the variable setting is visible.

To delete an existing variable, select the appropriate variable, and then click **Delete**. Do not delete any of the pre-defined variables. Before deleting a variable that you defined, make sure that none of your applications require the configuration attribute setting that the variable provides.

Name

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root used by WebSphere Application Server.

Value

Specifies the value that the symbolic name represents. For example, the value might be an absolute path value for a file or URL root.

Scope

Specifies the level at which a WebSphere variable is visible on the administrative console panel. The scope is specified when a new variable is defined.

A resource can be visible in the administrative console collection table at the node or server scope.

On a multiple-server product, a resource also can be visible at the cell or cluster scope. The cluster scope is only available if a cluster is defined for the cell.

WebSphere variables settings

Use this page to define the name and value of a WebSphere variable. A WebSphere variable is a name and value pair that is used to provide the setting for one of the string data type attributes contained in one of the XML formatted configuration files that reside in the product repository.

To view this administrative console page, click **Environment > WebSphere variables > WebSphere_variable_name**.

Name:

Specifies the symbolic name for a product variable. After the variable is defined, this symbolic name can be specified in the **Value** field of any other product configuration field that accepts a string value. Whenever the application server encounters a configuration field that contains one or more symbolic names, it replaces the symbolic names with their defined values. For example, you might define a variable name that represent a commonly used file path or URL.

WebSphere Application Server variables are used for:

- Configuring WebSphere Application Server path names, such as *JAVA_HOME*, and *APP_INSTALL_ROOT*.
- Configuring certain cell-wide customization values.
- Configuring the z/OS location service for the product.

For example, *WAS_SERVER_NAME* is the pre-defined symbolic name of the variable that represents the name of the default application server that is provided with the product.

Value:

Specifies the value that the symbolic name represents.

For example, *server1* is the value of a pre-defined variable *WAS_SERVER_NAME*.

Data type String

Description:

Documents the purpose of a variable.

Data type String

Introduction: Variables

Variables come in many varieties. They are used to control settings and properties relating to the server environment. The three main types of variables that you should understand are environment variables, WebSphere variables, and custom properties.

Environment variables. Environment variables, also called *native environment variables*, are not specific to WebSphere Application Server and are defined by other elements, such as UNIX, Language Environment® (LE), or third-party vendors, among others. Some of the UNIX-specific native variables are *LIBPATH* and *STEPLIB*. These variables tend to be operating system-specific.

Environment variables can also be specified as a servant custom property. To specify an environment variable as a servant custom property, in the administrative console, click **Servers > Server Types > WebSphere application servers***server_name*. Then, under Server Infrastructure, click **Java process management > Process definition**, select either **Control**, **Servant**, or **Adjunct**, and then click

Environment entries. This path is also used to set environment variables that control the collection of application server and Web container information in z/OS System Management Facility (SMF) records.

WebSphere variables

WebSphere variables are name and value pairs that are used to provide settings for any of the string data type attributes contained in one of the XML formatted configuration files that reside in the product repository. After a variable is defined, the value specified for the variable replaces the variable name whenever the variable name is encountered during configuration processing.

WebSphere variables can be used to configure:

- WebSphere Application Server path names, such as JAVA_HOME, and APP_INSTALL_ROOT
- Certain cell-wide customization values
- The location service for the z/OS platform.

To create or modify a WebSphere variable, in the administrative console click **Environment > WebSphere variables**.

A variable can apply to a cell, a cluster, a node, or a server.

How the variable is set determines its scope. If the variable is set:

- At the server level, it applies to the entire server.
- At the node level, it applies to all servers in the node, unless you set the same variable at the server level. In that case, for that server, the setting that is specified at the server level overrides the setting that is specified at the node level.
- At the cell level, it applies to all nodes in that cell, unless you set the same variable at the node or server level.
 - If you set the same variable at the server level, for that server, the setting that is specified at the server level overrides the setting that is specified at the cell level.
 - If you set the same variable at the node level, for all servers in that node, the setting that is specified at the node level overrides the setting that is specified at the cell level.

Custom properties

Custom properties are property settings meant for a specific functional component. Any configuration element can have a custom property. Common configuration elements are cell, node, server, Web container, and transaction service. A limited number of supported custom properties are available and these properties can be set in the administrative console using the custom properties link that is associated with the functional component.

For example, to set Web container custom properties, click **Servers > Server Types > WebSphere application servers > *server_name***, and then, in the Container settings section, click **Web container > Custom properties**

Custom properties set from the Web container custom properties page apply to all transports that are associated with that Web container; custom properties set from one of the Web container transport chain or HTTP transport custom properties pages apply only to that specific HTTP transport chain or HTTP transport. If the same property is set on both the Web container page and either a transport chain or HTTP transport page, the settings on the transport chain or HTTP transport page override the settings that are defined for the Web container for that specific transport.

Note: You can only specify custom properties for an HTTP transport that is being used by an application server that is running on a Version 5.1.x node in a mixed cell environment.

WebSphere Variables

WebSphere variables are name and value pairs that are used to provide settings for any of the string data type attributes that are used to configure the product. After a variable is defined, the symbolic name that is specified for that variable can be specified in the **Value** field of any other configuration field for the product that accepts a string value.

When a variable is defined, it is given a scope. The scope is the range of locations within the product network where the variable is applicable.

- A variable with a cell-wide scope is available across the entire deployment manager cell.
- A variable with a cluster-wide scope is available across the entire cluster in the cell.
- A variable with a node-level scope is available only on the node and the servers on that node. If a node-level variable has the same name as a cell-wide variable, the node-level variable value takes precedence.
- A server variable is available only on the one server process. A server variable takes precedence over a variable with the same name that is defined at a higher level.

The value of a configuration attribute can contain references to one or more variables. The syntax for such an attribute is the name of the variable, enclosed in either a pair of curly braces { } or a pair of parenthesis (). In either case, the variable is preceded by the dollar sign.

A string configuration attribute value can consist of:

- String literals, including the null value and an empty string
- Variable references that each includes one or more levels of indirection
- Nested variable references.
- Any combination of non-null and non-empty string literals, variable references, and nested variable references.

The following table illustrates all of the possible combinations.

Table 2.

Configuration attribute consists of:	Configuration attribute value	Variable name	Second variable value	Third variable value	Fourth variable value	Expanded configuration attribute value
String literal	/IBM/ WebSphere/ AppServer	N/A	N/A	N/A	N/A	/IBM/ WebSphere/ AppServer
Variable reference	\$(WAS_ INSTALL_ ROOT)	WAS_ INSTALL_ ROOT	/IBM/ WebSphere/ AppServer	N/A	N/A	/IBM/ WebSphere/ AppServer
Variable reference with a string literal	\$(USER_ INSTALL_ ROOT)/temp	USER_ INSTALL_ ROOT	N/A	N/A	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01/temp
Indirect variable reference with a string literal	\$(WAS_ INSTALL_ ROOT)/lib	WAS_ INSTALL_ ROOT	\$(MY_ INSTALL_ ROOT)	MY_ INSTALL_ ROOT	N/A	N/A
Nested variable references with string literal (Example 1)	\$(\${INSTALL_ TYPE}_ INSTALL_ ROOT)/lib	INSTALL_ TYPE	USER	USER_ INSTALL_ ROOT	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01	/IBM/ WebSphere/ AppServer/ profiles/ AppSrv01/lib
Nested variable references with string literal (Example 2)	\$(\${INSTALL_ TYPE}_ INSTALL_ ROOT)/lib	INSTALL_ TYPE	WAS	WAS_ INSTALL_ ROOT	/IBM/ WebSphere/ AppServer/ AppServer	/IBM/ WebSphere/ AppServer/ AppServer/lib

During the configuration process, whenever a variable is encountered as the value for a configuration attribute, a variable expansion is performed on that variable. A variable expansion is the process of recursively replacing variable references with variable values until only a string literal remains as the value for the configuration attribute. If the expansion process encounters a variable that is not properly defined, the expansion of that variable stops and a `VariableExpansionException` exception is issued. The product configuration process continues. However, processing errors might occur because the value for this configuration attribute is not properly established.

Note: The variable expansion syntax that is provided in Versions 5.1.x, 6.0.x, and 6.1.x, of the product, includes a variant that consists of a dollar sign, and a single letter variable name without any surrounding braces or parenthesis. This syntax is not supported in Version 7.0 or higher. All WebSphere variables references must be surrounded by matching parenthesis or braces, even if it is a single letter. That syntax required escaping of dollar signs to avoid ambiguity. For backward compatibility, the escaping of the literal dollar sign is still supported, and the literal dollar sign is interpreted as indicated in the following table.

Table 3.

Input value	Value after expansion
\$	\$
\$\$	\$
\$\$\$	\$\$
\$\$\$\$	\$\$
\$\$\$\$\$	\$\$\$

Configuring the IBM Toolbox for Java

The IBM Toolbox for DB2® is a library of Java classes that are optimized for accessing i5/OS data and resources. You can use the IBM Toolbox for Java JDBC driver to access local or remote DB2 UDB for iSeries® databases from server-side and client Java applications that run on any platform that supports Java.

Before you begin

Determine which version of the IBM Toolbox for Java you want to use on your system.

About this task

The IBM Toolbox for Java is available in these versions:

IBM Toolbox for Java licensed program

The licensed program is available with every i5/OS release. You can install the licensed program on your i5/OS system, and then either copy the IBM Toolbox for Java JAR file (*jt400.jar*) to your system or update your system *classpath* to locate the server installation. Product documentation for IBM Toolbox for Java is available from the i5/OS information center: <http://publib.boulder.ibm.com/infocenter/series/v5r4/index.jsp> Locate the documentation by traversing the following path in the left-hand navigation window of the iSeries information center:

Programming > Java > IBM Toolbox for Java.

JTOpen

JTOpen is the open source version of IBM Toolbox for Java, and is more frequently updated than the licensed program version. You can download JTOpen from <http://www.ibm.com/servers/eserver/series/toolbox/downloads.htm>. You can also download the *JTOpen Programming Guide*. The guide includes instructions for installing JTOpen and information about the JDBC driver.

The IBM Toolbox for Java JDBC driver is included with both versions of the IBM Toolbox for Java. This JDBC driver supports JDBC 3.0. For more information about IBM Toolbox for Java and JTOpen, see the product Web site at <http://www.ibm.com/servers/eserver/series/toolbox/index.html>.

Note: If you are using the product on platforms other than iSeries, use the **JTOpen** version of the Toolbox JDBC driver.

1. Download the *jt400.jar* file from the **JTOpen** URL at <http://www.ibm.com/servers/eserver/series/toolbox/downloads.htm>.

Place it in a directory on your workstation such as `/JDBC_Drivers/Toolbox`.

2. Open the administrative console.
3. Select **Environment > WebSphere variables**.
4. In the list of available scopes, select the appropriate node.
5. Locate the WebSphere variable `OS400_TOOLBOX_JDBC_DRIVER_PATH` in the list of variables that are defined for that scope.

Depending on how many variables are defined for the selected node, you might have to navigate through multiple pages of variables to find the `OS400_TOOLBOX_JDBC_DRIVER_PATH` variable. In this situation, clicking the arrow at the bottom of the page takes you to the next page of variables for the selected node.

6. Click **OS400_TOOLBOX_JDBC_DRIVER_PATH** in the name column.
7. Set the value to the full directory path to the *jt400.jar* file downloaded in step one. Do not include *jt400.jar* in this value.

For example, if the fully qualified path to the *jt400.jar* file is:

```
JDBC_Drivers/Toolbox/jt400.jar
```

Specify `JDBC_Drivers/Toolbox` as the value for the `OS400_TOOLBOX_JDBC_DRIVER_PATH` variable.

8. Click **Apply** and then click **Save** to save your changes.

Repository service custom properties

Use this page to add custom properties for the repository service.

You can specify repository service custom properties in the administrative console:

1. In the administrative console navigation, click **System Administration > Node agents**.
2. Select a node agent from the list.
3. Under Additional Properties, click **File synchronization service**.
4. Under Additional Properties, click **Custom properties**.
5. Click **New**.
6. Enter the name of the custom property in the Name field, and the value in the Value field. You can leave the Description field blank.

recoveryNode

Specifies that a node is a recovery node. This property is only supported for a z/OS environment.

Set this value to `true` if you want a node in a Network Deployment cell to act as a peer restart and recovery node for another node in the same cell. The recovery node shadows the complete configuration of its recovery peer. Use this property if you need to support peer restart and recovery only and are not using a shared file system.

Data type

Boolean

Application server custom properties for z/OS

Some of the application server custom properties that are provided with the product can only be used with z/OS. This topic describes how to use these properties.

Note: Setting these custom properties at the server level is deprecated. However, you can specify them as WebSphere variables with a scope of either a specific server, specific node, or specific cell. Server scoped WebSphere variables still override any settings specified at the node scope, or higher, and are added to the was.env file.

To set one of these custom properties for either an application server or a deployment manager, in the administrative console, click **Environment > WebSphere variables**, select the appropriate node or cell from the list of available servers, nodes and cells, and then click **New**.

adjunct_jvm_direct_options

Specifies options that you need to pass directly to the Java virtual machine (JVM) launch in the adjunct. This property is typically used for JVM options that the JVM cannot read from the options file that is specified as the value of the control_region_jvm_properties_file property. For example, the JVM cannot read the value that is specified for the -memorycheck option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the servant_jvm_direct_options and control_jvm_direct_options custom properties to specify options that you need to pass directly to the JVM launch in the servant and controller, respectively.

Data Type	String
Default	Empty string
Used by Daemon	No

control_region_dreg_on_no_srs

Specifies whether the controller rejects requests for dispatch within a servant when it detects that no servants are available to process requests.

When this property is set to 1, if the controller detects that there are no servants available to process requests, it rejects requests for dispatch within the servants. It also removes the application server from the registry of servers that workload management (WLM) uses to assign work, and stops the HTTP and message-driven bean (MDB) listeners. If this property is set to 0, then the function is disabled.

When the minimum number of servants become available, the controller registers the application server with WLM again, starts the HTTP and MDB listeners, and allows requests to be dispatched to the servants.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_confirm_recovery_on_no_srs

Specifies whether requests are dispatched to servants following the detection of a no-servants situation. This property is ignored if the control_region_dreg_on_no_srs custom property is set to 0.

When this property is set to 1, the controller does not dispatch requests to the servants until it receives a response to message BBOO0297A. This message is issued following a no-servant situation when the sever detects that the required minimal number of servants are available to process requests.

When this property is set to 0 (zero), the controller determines when to allow requests to be dispatched to the servants after a no-servant condition is detected.

Data Type	Integer
Acceptable values	0 or 1
Default	0

control_region_http_queue_timeout_percent

Specifies the percentage of the HTTP dispatch time limit that is used as the maximum amount of time that an HTTP request can spend on the workload management (WLM) queue. The `protocol_http_timeout_output` custom property is used to specify the maximum amount of time that an HTTP request can spend on the queue and in dispatch before an error message is issued, which indicates that an HTTP dispatch timeout has occurred.

Data Type	Integer
Range	0 - 99
Default	0

control_region_https_queue_timeout_percent

Specifies the percentage of the HTTPS dispatch time limit that is used as the maximum amount of time that an HTTPS request can spend on the workload management (WLM) queue. The `protocol_https_timeout_output` custom property is used to specify the maximum amount of time that an HTTPS request can spend on the queue and in dispatch before an error message is issued, which indicates that an HTTPS dispatch timeout has occurred.

Data Type	Integer
Range	0 - 99
Default	0

control_region_iiop_queue_timeout_percent

Specifies the percentage of the IIOp dispatch time limit that is used as the maximum amount of time that an IIOp request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the `control_region_wlm_dispatch_timeout` custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

Data Type	Integer
Range	0 - 99
Default	0

control_region_mdb_queue_timeout_percent

Specifies the percentage of the MDB dispatch time limit that is used as the maximum amount of time that an MDB request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the `control_region_mdb_request_timeout` custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

Data Type	Integer
Range	0 - 99
Default	0

control_region_mdb_request_timeout

Specifies the time, in seconds, that the server waits for a message-driven bean (MDB) request to receive a response. If the response is not received within the specified amount of time, the server removes the MDB request, and issues an error message that indicates that an MDB dispatch timeout has occurred.

Set this value to 0 to disable the function.

Data Type	Integer
Units	Seconds
Default	120

control_region_sip_queue_timeout_percent

Specifies the percentage of the Session Initiation protocol (SIP) dispatch time limit that is used as the maximum amount of time that a SIP request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the `protocol_sip_timeout_output` custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

Data Type	Integer
Range	0 - 99
Default	0

control_region_sips_queue_timeout_percent

Specifies the percentage of the SIP Secure Sockets Layer (SSL) dispatch time limit that is used as the maximum amount of time that a SIP SSL request can spend on the workload management (WLM) queue.

This property only applies to the time that the request spends on the WLM queue. Use the `protocol_sips_timeout_output` custom property if you want to limit the amount of time that the request spends on both the WLM queue and in dispatch,

Data Type	Integer
Range	0 - 99
Default	0

control_region_timeout_delay

Specifies the number of seconds a controller waits after detecting a timeout before it terminates the servant. This time delay gives work that is currently running in the servant a chance to complete before the servant is terminated.

The specified length of time period starts when a timeout occurs. When a servant thread completes its current work item and determines that the servant is being terminated, the servant thread waits for the specified length of time instead of selecting a new work item.

When this field is set to 0 the controller terminates a servant as soon as the controller detects a timeout.

Data Type	Integer
Units	Seconds
Default	0

This property is affected by the setting for the `server_use_wlm_to_queue_work` property:

- If the `server_use_wlm_to_queue_work` property is set to 1, during the time period specified for the `control_region_timeout_delay` property, exactly one new request can be processed by each servant worker thread that is idle when the timeout occurred.

Only one new request is processed because, when the timeout occurs, idle servant worker threads are paused waiting for WLM to assign new to them. When WLM assigns new work to one of these threads, the thread becomes active, processes the work, and then acquiesces. Therefore, WLM cannot assign any additional work to this thread.

- If the `server_use_wlm_to_queue_work` property is set to 0, during the time period specified for the `control_region_timeout_delay` property, work requests that were not yet dispatched but were queued without affinity to the terminating servant, are requeued to another available servant after the servant termination process completes.

control_region_timeout_dump_action

Specifies the type of dump that is taken whenever a timeout occurs for work that has been dispatched to a servant. This property only applies if the `control_region_timeout_delay` custom property is set to a non-zero value.

Set this property to either `JAVACORE` or `javacore`, if you want a Java core dump, or set the property to either `SVCDUMP` or `svcdump` if you want an SVC dump.

Default None

control_region_timeout_dump_action_session

Specifies the type of dump that is taken whenever a timeout occurs for an HTTP, HTTPS, SIP, or SIPS request that has been dispatched to a servant.

This property only applies if the following corresponding variable is set to `SESSION`:

- protocol_http_timeout_output_recovery
- protocol_https_timeout_output_recovery
- protocol_sip_timeout_output_recovery
- protocol_sips_timeout_output_recovery

The value specified for this property determines whether a Java core dump or an SVC dump is taken. Set this property to either `JAVACORE` or `javacore`, if you want a Java core dump, or set the property to either `SVCDUMP` or `svcdump` if you want an SVC dump.

Default None

controller_jvm_direct_options

Specifies options that you need to pass directly to the JVM launch in the controller. This property is typically used for JVM options that the JVM cannot read from the options file that is specified as the value of the `control_region_jvm_properties_file` property. For example, the JVM cannot read the value that is specified for the `-memorycheck` option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the `servant_jvm_direct_options` and `adjunct_jvm_direct_options` custom properties to specify options that you need to pass directly to the JVM launch in the servant and adjunct, respectively.

Data Type String
Default Empty string
Used by Daemon No

control_region_timeout_save_last_servant

Specifies whether the controller terminates the last available servant when a timeout situation occurs. If the controller does not terminate the last available servant when a timeout situation occurs, other work continues to be processed until a new servant is initialized. However, not terminating the last available servant might cause the loss of system resources if the dispatched servant thread that encountered the timeout situation continues to loop or stops functioning. For example, if timeouts keep occurring, the system might consume a high percentage of the available servant threads.

The functionality of this property depends on the values that you specify for other custom properties:

- If you set this property to 1, and the `wlm_minimumSRCount` custom property is set to a value that is greater than 1, when a timeout situation occurs, the controller waits until a new servant is initialized before it terminates the last available servant.
- If you set this property to 0, or when a timeout situation occurs, the controller terminates the last available servant. It does not wait for another servant to be initialized.
- If the `wlm_dynapplenv_single_server` custom property is set to 1, the value that you specify for this property is ignored.

Data Type	Boolean
Default	0

default_internal_work_transaction_class

Specifies the default transaction class for internally processed work within the server.

If an internal classification element is not listed in the `wlm_classification_file`, or if the `wlm_classification_file` is not specified, then the `default_internal_work_transaction_class` setting is used. If a value is specified for internal classification setting listed in the `wlm_classification_file`, The value specified for the `default_internal_work_transaction_class` custom property is ignored.

Data Type	String
Default	null (empty string)
Used by Daemon	No

iiop_max_msg_megsize

Specifies, in megabytes, the maximum size for IIOP requests. For example, if you set the property to 35, then any requests over 35 MB are rejected. The minimum value for this property is 10, and the maximum value is 2048. Omit this property if you do not want to limit the size of IIOP requests.

Note: This custom property only applies to application servers that are running in 64-bit mode. The maximum size for IIOP requests that are being handled by application servers running in 31-bit mode is 10 MB.

Data Type	Integer
Default	0
Used by Daemon	No

local_comm_max_msg_megsize

Specifies, in megabytes, the maximum size of locally connected communications requests. For example, if you set the property to 35, then any requests over 35 MB are rejected. The minimum value for this property is 10 and the maximum value is 2048. Omit this property if you do not want to limit the size of locally connected communications requests.

Note: This custom property only applies to application servers that are running in 64-bit mode. The maximum size for IIOP requests that are being handled by application servers running in 31-bit mode is 10 MB.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_accept_http_work_after_min_srs

Specifies whether the application server waits until a minimum number of servants are ready to accept work before the application server starts the HTTP transport channels. If this property is set to `true`, then when the minimum number of servants are ready for work, the HTTP transport channels start accepting work. If this property is set to `false`, the HTTP transport channels start when the controller starts.

When this property is set to `true`, the value specified for the Minimum number of instances property determines the number of servants that must be ready before the HTTP transport channels start. To change the setting of the Minimum number of instances property for an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Java and process management > Server instance**. To change the setting of this property for a deployment manager, in the administrative console, click **System Administration > Deployment manager > Java and process management > Server instance**.

The job output indicates `protocol_accept_http_work_after_min_srs: 1`, if this property is set to `true`, or `protocol_accept_http_work_after_min_srs: 0`, if this property is set to `false`.

Data Type	Boolean
Default	<code>true</code>
Used by Daemon	No

protocol_accept_iiop_work_after_min_srs

Specifies whether the application server waits for a minimum number of servants to be ready to accept work before the application server starts the IIOp transport channels. If this property is set to `1`, when the minimum number of servants is ready for work, then the IIOp transport channels starts accepting work. If this property is set to `0`, the IIOp transport channels start when the controller starts.

When this property is set to `true`, the value specified for the Minimum number of instances property determines the number of servants that must be ready before the IIOp transport channels start. To change the setting of the Minimum number of instances property for an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Java and process management > Server instance**. To change the setting of this property for a deployment manager, in the administrative console, click **System Administration > Deployment manager > Java and process management > Server instance**.

Data Type	Boolean
Default	0
Used by Daemon	No

protocol_bboc_log_response_failure

Specifies that if the BBOO0168W message is issued, then the failure that is detected when attempting to send a response to a client is recorded. The message is sent to the error log. The message text contains the request method name, the reply status, and the routing information that identifies the client.

Data Type	Boolean
Default	<code>false</code>
Used by Daemon	Yes

protocol_bboc_log_return_exception

Specifies that if the BBOO0169W message is issued, the response that contains the SystemException is recorded. The message is sent to the error log. The message text contains the exception identifier and minor code, the request method name, and the routing information that identifies the client.

Note: This property only applies to application servers. This property is ignored if it is specified for a deployment manager.

Data Type	Boolean
Default	false
Used by Daemon	Yes

protocol_giop_level_highest

Specifies the CORBA General Inter-ORB Protocol (GIOP) protocol version level that is used by the application server object request broker (ORB). Valid values are 1.1 and 1.2. Interoperable object references (IORs) that are exported from this server use the GIOP level indicated.

You might need to change the setting of this property from the default value if you use a client ORB that is not shipped as part of the product, and that supports a previous version of the CORBA standard. For example, you might need to change from the default protocol version level of 1.2 to 1.1 to support a client ORB that supports the 1.1 CORBA standard instead of the 1.2 CORBA standard.

Note: The maximum GIOP level that the daemon address space supports is 1.1. The GIOP level has of the daemon has no effect on the GIOP levels of the application servers that connect to the daemon.

Data Type	String
Default	1.2
Used by Daemon	Yes

protocol_http_backlog

Specifies the maximum length for the queue of pending connections using HTTP. The value that you specify can be limited by the specification of the SOMAXCONN statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_http_large_data_inbound_buffer

Specifies, in bytes, the limit of the size of incoming HTTP requests when inbound HTTP chunking is disabled. For example, if you set the property to 15728640, any requests over 15 MB are rejected. Specify 0 to reject any requests that are larger than 10 MB.

Note: Use this custom property only for an application server that is running in 31-bit mode. If the application server is running in 64-bit mode, then use the protocol_http_large_data_inbound_buffer_64bit custom property to set a limit for this inbound buffer.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_large_data_inbound_buffer_64bit

Specifies, in megabytes, the size limit for incoming HTTP requests when inbound HTTP chunking is disabled. For example, if you set the property to 35, any HTTP requests over 35 MB are rejected. Specify 0 for this property if you do not want to limit the size of unchunked HTTP requests.

Note: Use this custom property only for an application server that is running in 64-bit mode. If the application server is running in 31-bit mode, then use the `protocol_http_large_data_inbound_buffer` custom property to set a limit for this inbound buffer.

Data Type	Integer
Default	0
Used by Daemon	No

protocol_http_large_data_response_buffer

Specifies, in bytes, the maximum length of the response buffer that is used for HTTP requests. Responses larger than this value are rejected. Specify a value of 0 if you do not need a large response buffer because all of your HTTP responses are less than 10 MB.

Note: This property only applies to application servers. It does not apply to a deployment manager.

Data Type	Integer
Default	104857600
Used by Daemon	No

protocol_http_resolve_foreign_hostname

Specifies whether to perform Domain Name Server (DNS) resolution of the IP address of a foreign client to a DNS registered host name for each established HTTP session. If this property is set to 1, then the DNS host name resolution is performed. If this property is set to 0, then the DNS host name resolution is not performed, and a textual representation of the IP address of the foreign client is used instead of the DNS host name.

Data Type	Boolean
Default	0
Used by Daemon	Yes

protocol_http_timeout_output_recovery

Specifies the recovery action that is taken when an HTTP request does not complete within a designated length of time. Setting this property to `SERVANT` allows servants to terminate when a timeout occurs. If an HTTP request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The HTTP request and socket are then cleaned up. If this property is set to `SESSION`, no attempt is made to disrupt the processing of a dispatched HTTP request within a servant. However, the HTTP request and socket are still cleaned up. Using the `SESSION` setting might result in a loss of resources if the dispatched HTTP request loops or becomes inactive.

Default	SERVANT
Used by Daemon	No

protocol_https_backlog

Specifies the maximum length for the queue of pending connections using HTTPS. The value that you specify can be limited by the specification of the `SOMAXCONN` statement in the TCP/IP Profile.

Data Type	Integer
Default	10
Used by Daemon	No

protocol_https_cert_mapping_file

Specifies the name of a file containing entries that map IP addresses to server certificate labels. You can set this property at the cell, node, or server level.

Note: The `protocol_https_cert_mapping_file` property is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

When an HTTP SSL connection request is received, the application server checks the IP address against entries in the file specified for this property. If the application server finds a match, the certificate mapped to the IP address is used for the connection. If the application server does not find a match, it checks the `protocol_https_default_cert_label` property for the name of a certificate. If a certificate name is specified, the application server uses that certificate to establish the connection. If a certificate name is not specified, the default server certificate specified in the RACF SSL keyring, that is owned by the application server, is used to establish the HTTP SSL connection.

protocol_https_default_cert_label

Specifies the label of the server certificate that the application server uses when establishing HTTP SSL connections with the application server. You can set this property at the cell, node, or server level.

Note: The `protocol_https_default_cert_label` property is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

If the name of a certificate is not specified for this property, the default server certificate specified in the RACF SSL keyring, that is owned by the application server, is used to establish the HTTP SSL connection.

protocol_https_timeout_output_recovery

Specifies the recovery action that is taken when an HTTPS request does not complete within a designated length of time. Setting this property to `SERVANT` allows servants to terminate when a timeout occurs. If an HTTPS request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The HTTPS request and socket are then cleaned up. If this property is set to `SESSION`, no attempt is made to disrupt the processing of a dispatched HTTPS request within a servant. However, the HTTPS request and socket are still cleaned up. Using the `SESSION` setting might result in a loss of resources if the dispatched HTTPS request loops or becomes inactive.

Default	SERVANT
Used by Daemon	No

protocol_iiop_backlog

Specifies the maximum length for the queue of pending connections using the CORBA Internet Inter-ORB protocol (IIOP). The value that you specify might be limited by the specification of the `SOMAXCONN` statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_backlog_ssl

Specifies the maximum length for the queue of pending connections using IIOP Secure Sockets Layer (SSL). The value that you specify can be limited by the specification of the `SOMAXCONN` statement in the TCP/IP profile.

Data Type	Integer
Default	10
Used by Daemon	Yes

protocol_iiop_local_propagate_wlm_enclave

Specifies whether to propagate the workload management (WLM) enclave that is associated with the currently dispatched request on an outbound IIOp request that was made to another server on the same z/OS system over local interaddress space communication protocols.

Data Type	Boolean
Default	1
Used by Daemon	No

protocol_iiop_resolve_foreign_hostname

Specifies whether to perform Domain Name Server (DNS) resolution of the IP address of a foreign client to a DNS registered host name for each established IIOp session. If this property is set to 1, then the DNS host name resolution is performed. If this property is set to 0, then the DNS host name resolution is not performed, and a textual representation of the IP address of the foreign client is used instead of the DNS host name.

Data Type	Boolean
Default	1
Used by Daemon	Yes

protocol_sip_timeout_output_recovery

Specifies the recovery action that is taken when a SIP request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If a SIP request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The SIP request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched SIP request within a servant. However, the SIP request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched SIP request loops or becomes inactive.

Default	SERVANT
Used by Daemon	No

protocol_sips_timeout_output_recovery

Specifies the recovery action that is taken when a SIPS request does not complete within a designated length of time. Setting this property to SERVANT allows servants to terminate when a timeout occurs. If a SIPS request is under dispatch in a servant when its timeout value is reached, the servant terminates with an ABEND EC3 RSN=04130007. The SIPS request and socket are then cleaned up. If this property is set to SESSION, no attempt is made to disrupt the processing of a dispatched SIPS request within a servant. However, the SIPS request and socket are still cleaned up. Using the SESSION setting might result in a loss of resources if the dispatched SIPS request loops or becomes inactive.

Default	SERVANT
Used by Daemon	No

ras_debugEnabled

Specifies to use an external debugger tool with the application server for tracing and debugging client and server application components such as JavaServer Pages (JSP) files, servlets, and enterprise beans.

Data Type	Boolean
------------------	---------

Default	false
Used by Daemon	Yes

ras_default_msg_dd

Specifies whether to redirect write-to-operator (WTO) messages that use the default routing to SYSPPRINT. These messages are redirected to the location identified through the DD card on the JCL start procedure for the server. These WTO messages are typically issued during initialization.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_dumpoptions_dumptype

Specifies the default dump that is used by the signal handler. Do not change this property unless directed to do so by IBM Support personnel.

0	No dump is generated.
1	A ctrace dump is taken.
2	A cdump dump is taken.
3	A csnap dump is taken.
4	A CEE3DMP dump is taken.

Data Type	Integer
Default	3
Used by Daemon	Yes

ras_dumpoptions_ledumpoptions

Specifies the dump options to use with a CEE3DMP dump. If you want more than one option, then separate each option with a blank space. Do not change this property unless directed to do so by IBM service personnel.

Data Type	String
Default	THREAD(ALL) BLOCKS
Used by Daemon	Yes

ras_hardcopy_msg_dd

Specifies to redirect write-to-operator (WTO) messages that are routed to hard copy. These messages are redirected to the location that is identified through the DD card on the server JCL start procedure. These WTO messages are primarily audit messages that are issued from Java code during initialization.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_log_logstreamName

Specifies the log stream that the product uses for error information. If the specified log stream is not found or not accessible, a message is issued and errors are written to the server job log. If this variable is not specified, the product uses the SYSOUT stream.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_minorcode_action

Specifies the default behavior for gathering documentation about system exception minor codes.

Data Type	String
Default	NODIAGNOSTICDATA
Used by Daemon	Yes

You can also specify the following values.

CEEDUMP	Captures callback and offsets. Taking a CEE dumps is a lengthy process, and transaction time outs can occur during this process.
TRACEBACK	Captures Language Environment and UNIX traceback data for the z/OS operating system.
SVCDUMP	Captures an MVS dump, but does not produce a dump in the client.

ras_stderr_ff_interval

Specifies the interval of time, in minutes, between the writing of a form-feed character to standard error (SYSOUT).

Note: Combine this environment setting with the SEGMENT= parameter on the SYSOUT DD statement. The value of the SEGMENT= parameter is the number of form-feeds that are required before the first segment is closed and a new segment is allocated.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_stderr_ff_line_interval

Specifies the number of lines of output that is written between the writing of form-feed characters to standard error (SYSOUT).

Because of uncontrollable factors, such as line wrapping, the product can only approximate the number of lines of output it has written. Therefore the actual number of lines written between the writing of form-feed characters might be plus or minus 5 percent of the value specified for the property.

Note: Combine this environment setting with the SEGMENT= parameter on the SYSOUT DD statement. The value of the SEGMENT= parameter is the number of form-feeds that are required before the first segment is closed and a new segment is allocated.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_stdout_ff_interval

Specifies the interval of time, in minutes, between the writing of a form-feed character to standard output(SYSPPRINT).

Note: Combine this environment setting with the SEGMENT= parameter on the SYSPPRINT DD statement to segment the output. The value of the SEGMENT= parameter is the number of form-feeds that are required before the first segment is closed and a new segment is allocated.

Data Type	Integer
------------------	---------

Default	0
Used by Daemon	Yes

ras_stdout_ff_line_interval

Specifies the number of lines of output between the writing of form-feed characters to standard output (SYSPRINT).

Because of uncontrollable factors, such as line wrapping, the product can only approximate the number of lines of output it has written. Therefore the actual number of lines written between the writing of form-feed characters might be plus or minus 5 percent of the value specified for the property.

Note: Combine this environment setting with the SEGMENT= parameter on the SYSPRINT DD statement. The value of the SEGMENT= parameter is the number of form-feeds that are required before the first segment is closed and a new segment is allocated.

Data Type	Integer
Default	0
Used by Daemon	Yes

ras_time_local

Specifies whether time stamps in the error log display is in local time or Greenwich Mean Time (GMT). The time stamp is in GMT if this property is set to false.

Data Type	Boolean
Default	false
Used by Daemon	Yes

ras_trace_basic

Specifies tracing overrides for particular product subcomponents. Subcomponents, specified by numbers, receive basic and exception traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Do not change this property unless directed to do so by IBM service personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_trace_BufferCount

Specifies the number of trace buffers to allocate.

Data Type	Integer
Valid values	4 through 8
Default	4
Used by Daemon	Yes

ras_trace_BufferSize

Specifies, in bytes, the size of a single trace buffer. You can use the letters K, for kilobytes, or M, for megabytes.

Data Type	String
Valid values	128K through 4M
Default	1M
Used by Daemon	Yes

ras_trace_ctraceParms

Specifies the identify of the CTRACE PARMLIB member. The value can be either a two-character suffix, which is added to the CTIBBO string to form the name of the PARMLIB member, or the fully specified name of the PARMLIB member. For example, you can use the 01 suffix, which the system resolves to CTIBBO01. A fully specified name must conform to the naming requirements for a CTRACE PARMLIB member. For details, see z/OS MVS Diagnosis: Tools and Service Aids, GA22-7589.

If this property is specified and the PARMLIB member is not found, then the default PARMLIB member, CTIBBO00, is used. If neither the specified nor the default PARMLIB member is found, then tracing is defined to CTRACE, but no connection is available to a CTRACE external writer.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_trace_defaultTracingLevel

Specifies the default tracing level for the product. Use this variable with the ras_trace_basic and ras_trace_detail variables to set tracing levels for product subcomponents. Do not change this property unless directed by IBM Support personnel.

0	No tracing
1	Exception tracing
2	Basic and exception tracing
3	Detailed tracing, including basic and exception tracing

Data Type	Integer
Default	1
Used by Daemon	Yes

ras_trace_detail

Specifies tracing overrides for particular product subcomponents. Subcomponents, specified by numbers, receive detailed traces. If you specify more than one subcomponent, use parentheses and separate the numbers with commas. Do not change this property unless directed to do so by IBM Support personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

ras_trace_exclude_specific

Specifies product trace points to exclude from tracing activity.

Trace points are specified by 8-digit, hexadecimal numbers. Do not use this property unless directed to do so by IBM service personnel. If IBM service personnel directs you to specify more than one trace point, use parentheses and separate the numbers with commas. You also can specify a variable name by enclosing the name in single quotation marks.

Data Type	String
Default	Empty string
Used by Daemon	Yes

Note: Results sometimes depend on the value specified for the `ras_trace_minorCodeDefault` environment variable. If you specify `ras_trace_minorCodeTraceBacks=ALL` and `ras_minorcode_action=NODIAGNOSTICDATA`, you get a traceback. However, if you specify `ras_trace_minorCodeTraceBacks=(null value)` and `ras_minorcode_action=TRACEBACK`, you also get a traceback. Specifying `ras_trace_minorCodeTraceBacks=(null value)` causes TRACEBACK data to be collected, instead of cancelled.

ras_trace_outputLocation

Specifies where to send trace records. You can specify:

- `SYSPRINT`
- `BUFFER`, which sends the trace records to a memory buffer, the contents of which are later written to a CTRACE data set
- `TRCFILE`, which sends the trace records to the trace data set that is specified on the TRCFILE DD statement in the start procedure for the server.

For servers, you can specify one or more values, separated by a space.

Data Type	String
Default	SYSPRINT BUFFER
Used by Daemon	Yes

ras_trace_specific

Specifies tracing overrides for specific product trace points. Trace points are indicated by 8-digit, hexadecimal numbers. To specify more than one trace point, use parentheses and separate the numbers with commas. You can also specify tracing on a specific environment variable by using the name enclosed in single quotation marks. Do not use this property unless directed to do so by IBM Support personnel.

Data Type	String
Default	Empty string
Used by Daemon	Yes

register_ifaedreg_also

Specifies whether you want z/OS to create SMF Type 89 Subtype 2 records, in addition to SMF Type 89 Subtype 1 records. In previous releases of the product, after the product registered with z/OS, z/OS created SMF Type 89 Subtype 2 records to collect product usage data.

Note: Now, after the product registers with z/OS, z/OS produces SMF Type 89 Subtype 1 records instead of SMF Type 89 Subtype 2 records. If you want to have SMF Type 89 Subtype 2 records created in addition to the SMF Type 89 Subtype 1 records, add the `register_ifaedreg_also` variable to your WebSphere variables and set this property to 1. To turn off the creation of SMF Type 89 Subtype 2 records, set this variable to 0.

Data Type	Boolean
Default	0
Used by Daemon	No

security_SMF_record_first_auth_user

Specifies whether to record the first authenticated user under request dispatch in the SM120CRE field in the System Management Facility (SMF) server activity record.

If this property is set to 1, then the first authenticated user under request dispatch is written to the SM120CRE field. If this property is set to 0, then the ID of the user under which the server activity began is written to the SM120CRE field.

Data Type	Boolean
Default	0
Used by Daemon	No

servant_jvm_direct_options

Specifies options that you need to pass directly to the JVM launch in the servant. This property is typically used for Java virtual machine (JVM) options that the JVM cannot read from the options file that is specified as the value of the `control_region_jvm_properties_file` property. For example, the JVM cannot read the value that is specified for the `-memorycheck` option in the options file.

If you specify multiple options, use a semicolon to separate the options.

You can use the `controller_jvm_direct_options` and `adjunct_jvm_direct_options` custom properties to specify options that you need to pass directly to the JVM launch in the controller and adjunct, respectively.

Data Type	String
Default	Empty string
Used by Daemon	No

servant_region_custom_thread_count

Specifies the number of application threads that are used in each of the servants that are running in an application server.

If you specify a value for this custom property, you must set the Workload profile property on the ORB services z/OS additional settings page in the administrative console to **CUSTOM** before this setting becomes effective. To navigate to this page, in the administrative console, click **Servers > Application servers > *server_name* > Container services > ORB service > z/OS additional settings**.

Data Type	Integer
Range	1 - 100
Used by Daemon	No

server_region_http_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_https_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_iiop_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_mdb_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback
Used by Daemon	No

server_region_request_cputimeused_limit

Specifies, in milliseconds, the amount of CPU time that an application request can consume.

Note: The `server_region_request_cputimeused_limit` custom property helps you to prevent a single application request from monopolizing the available CPU time because it allows you to limit the amount of CPU time that a single request can use. A CPU monitor is invoked when a request is dispatched. If the request exceeds the specified amount of CPU time, the controller considers the request unresponsive. The controller then issues message BBOO0327, to let the requesting application know that the request was unresponsive.

The monitor, that monitors the amount of CPU time that a request is using, typically sends a signal to the dispatched thread when the amount of CPU time used exceeds the specified amount. However, there are situations when this signal cannot be delivered, and the request remains pending. For example, if the thread goes native and invokes a PC routine, the signal remains pending until the PC routine returns.

After the signal is delivered on the dispatch thread, the WLM enclave, that is associated with the dispatched request, is quiesced. This situation lowers the dispatch priority of this request, and this request should now only get CPU resources when the system is experiencing a light work load.

Data type	integer
Default	0
Used by Daemon	No

server_region_sip_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values	none, svcdump, javacore, heapdump, and traceback.
Default	traceback

Used by Daemon

No

server_region_sips_stalled_thread_dump_action

Specifies the type of dump that is taken when a request is considered unresponsive. After the dump is taken the controller is notified of the unresponsive request. The controller might then terminate the servant based on the values specified for other custom properties, such as `server_region_stalled_thread_threshold_percent`, `control_region_timeout_delay`, and `control_region_timeout_save_last_servant`.

Valid values

none, svcdump, javacore, heapdump, and traceback.

Default

traceback

Used by Daemon

No

server_region_stalled_thread_threshold_percent

Specifies the percentage of threads that can become unresponsive before the controller terminates the servant.

If 0 is specified, the controller terminates the servant as soon as the controller determines that at least one thread has become unresponsive.

Data Type

Integer

Default

0

Used by Daemon

No

server_SMF_request_activity_CPU_detail

Specifies whether you want the CPU usage breakdown section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the CPU usage breakdown section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the `server_SMF_request_activity_enabled` property is set to false.

Data Type

Boolean

Default

0

Used by Daemon

No

server_SMF_request_activity_enabled

Specifies whether you want the z/OS System Management Facility (SMF) to create an SMF 120 Subtype 9 record.

If you specify true for this property, an SMF 120 Subtype 9 record is created. Because this is a relatively large record, and collecting the data for this record could impact performance, do not enable this property unless you have a specific reason for collecting the data that is included in this record.

You can also lower the performance impact of creating this record by disabling one or more of the following properties.

- `server_SMF_request_activity_CPU_detail`
- `server_SMF_request_activity_security`
- `server_SMF_request_activity_timestamps`

Data Type

Boolean

Default

0

Used by Daemon No

server_SMF_request_activity_security

Specifies whether you want the Security data section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the Security data section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the `server_SMF_request_activity_enabled` property is set to false.

Data Type	Boolean
Default	0
Used by Daemon	No

server_SMF_request_activity_timestamps

Specifies whether you want the z/OS formatted timestamps section included in any SMF 120 Subtype 9 record that is created.

If this property is set to true, the z/OS formatted timestamps section is included in any SMF 120 Subtype 9 record that is created.

The setting for this property is ignored if the `server_SMF_request_activity_enabled` property is set to false.

Data Type	Boolean
Default	0
Used by Daemon	No

server_region_jvm_localrefs

Do not use this property unless directed to do so by IBM Support personnel.

Data Type	Integer
Default	128
Used by Daemon	No

server_region_jvm_logfile

Specifies the Hierarchical File System (HFS) file in which Java Native Interface (JNI) and class debug messages from the Java virtual machine (JVM) are logged. Use this variable only in a single-server environment. If you use this property in a multiple-server environment, then all of the servers write to the same file, and you might have difficulty using the file for diagnostic purposes.

Data Type	String (file name)
Default	Empty string
Used by Daemon	No

server_region_recycle_count

Specifies the number of transactions that are processed by a servant process after which the servant process is recycled. Workload management (WLM) ends the servant after all affinity requirements are met. Specify a nonzero value to enable recycling.

You might want to enable recycling if, after running for an extended period of time, your application is experiencing out-of-memory exceptions. Out-of-memory exceptions can result from memory leakage by your application.

Data Type	Integer
Default	0
Used by Daemon	No

server_start_wait_for_initialization_timeout

Specifies how long the startServer.sh command processing waits for the product initialization process to complete. By default, startServer.sh command processing waits indefinitely until initialization is complete.

Use this property if you want to complete one of the following actions:

- Control how long the application server waits for other dependent servers to start.
- Limit the amount of wait time when trying to debug problems with application initialization. For example, you might not want to wait if Web applications, that are automatically started, unexpectedly enter a longer than typical wait state.

Data Type	Integer
Default	0
Used by Daemon	No

com.ibm.ws.sib.ra.inbound.impl.MessageLockExpiry

When a message arrives on the queue that a message-driven bean (MDB) is consuming from, the message is locked and passed to the MDB in the servant region. If the servant region is disabled, or if there is an error processing the message on the servant region, this property defines how long the messaging engine waits before unlocking the message so that it can be re-delivered.

Data Type	Integer
Units	milliseconds
Default	300000
Range	A positive integer. The value 0 indicates that the message lock never expires and the messaging engine waits indefinitely for the servant region to process and unlock the message.

server_use_wlm_to_queue_work

Specifies whether WLM is used for workload queuing.

Set this property to 1 if you are using stateless application models. With these models, application objects, such as Enterprise JavaBeans™ (EJBs) and HTTP sessions, are only resident in memory for the life of an individual request. In this situation, you want WLM to dynamically balance individual requests. This configuration allows linear scalability and consistent, repeatable response times.

Set this property to 0 if you are using conversational application models. With these models, a client might hold, and periodically interact with, a reference to a stateful object which is pinned in the memory of one of the JVMs for a period of time that is greater than the duration of an individual request. For example, the client might be using HTTP sessions, stateful session beans or entity beans that are maintained in memory instead of being stored in a database or file system between requests, as is done in stateless application models.

Conversational application models prevent WLM from dynamically routing individual requests in a clustered environment because the client has affinity to a specific JVM. In this situation, a round robin algorithm is used to handle the initial request from the client. This algorithm evenly distributes the creation of long term affinities and is the best technique for achieving a balanced utilization of system resources in this type of environment. If you set this property to 0 for conversational application models, you must also set the server_work_distribution_algorithm property to 1.

If you prefer, you can exploit the round robin capability that WLM provides, instead of the previously described product round robin capability. The differences between the round robin capability that WLM provides and the product round robin capability is explained in the following scenario.

Note: A customer starts two clients to talk to a server. The server has two servants, and each servant has multiple threads. The customer expects one client to go to one servant, and the second client to go to the other servant. The behavior of product-provided round robin, that is initiated by specifying `server_use_wlm_to_queue_work=0`, and `server_work_distribution_algorithm=1`, adheres to this expectation. However, the WLM-provided round robin uses up all of the threads in the first servant before it starts to use the threads in the next servant. Therefore, in this situation, both clients go to the same servant and the second servant remains idle.

When the `server_use_wlm_to_queue_work` property is set to 0, the `wlm_minimumSRCCount` and `wlm_maximumSRCCount` properties are set to the same value. Because the work is not going through WLM, WLM only starts the number of servants that are specified for the `wlm_minimumSRCCount` property.

0	The z/OS WLM function is not used.
1	The z/OS WLM function is used.
Default	1
Used by Daemon	No

server_work_distribution_algorithm

Specifies the type of work distribution algorithm that the application server uses for workload balancing. This property is only used if the `server_use_wlm_to_queue_work` property is set to 0. If the `server_use_wlm_to_queue_work` property is set to 1, then the value specified for this property is ignored.

0	The hot thread algorithm is used. This option is not recommended.
1	The round robin algorithm is used. This value must be specified if the <code>server_use_wlm_to_queue_work</code> custom property is set to 0 for conversational application models.
Default	0
Used by Daemon	No

transaction_recoveryTimeout

Specifies the time, in minutes, that this controller uses to attempt to complete all restarted transactions before issuing a write-to-operator-with-reply (WTOR) message to the console, requesting whether to complete one of the following options:

- Continue the recovery process.
- Stop trying to resolve all restart transactions.
- Write transaction-related information to the job log or hard copy log.
- Terminate.

If the operator replies to continue the recovery process, then the controller attempts recovery for the specified amount of time before reissuing the write-to-operator message. After all the transactions are resolved, the controller terminates. This variable applies only to controllers that are running in peer restart and recovery mode.

Data Type	Integer
Default	15
Used by Daemon	No

wlm_classification_file

Specifies the location of your workload classification document. You can set this property at the cell, node, or server level.

The workload classification document is a common XML file that classifies inbound HTTP, IIOp, and message-driven bean (MDB) work. Any rules that are defined in this XML file override the old format HTTP classification. The rules in the common XML file also override any rules that are in the MDB classification file defined by the `endpoint_config_file` property.

For example, your configuration has the common XML file defined in a server that also has the old format HTTP classification document specified. There are no HTTP classification rules defined in your common XML file, so the old format file is used to classify inbound HTTP requests. However, if your common XML file contains HTTP rules, these classification rules are used instead of classification rules that you defined in the old style HTTP classification document.

wlm_servant_start_parallel

Specifies how a server, that is configured to start more than one servant address space, starts these address spaces.

- If you specify 1 for this property, then the server starts the first servant address space. After the first servant is completely initialized, the server then starts the remaining address spaces in parallel, which means that these remaining servant address spaces are all started at the same time.
- If you specify 0 for this property, then the server starts all of the address spaces sequentially, which means that a servant address space must be completely initialized before the next servant address space is started.

Note: This property requires z/WLM support. Therefore, before specifying this property, verify that you are running the product on z/OS Version 1, Release 9 or higher. If you are running the product on a lower level z/OS, specifying this property has no effect.

Data Type	Boolean
Default	0
Used by Daemon	No

wlm_stateful_session_placement_on

Specifies whether round robin queuing of HTTP sessions is enabled among servants. You can set this property at the cell, node, or server level.

Set this property to `true` if you want round robin queuing of HTTP sessions enabled among servants. Set this property to `false` if you do not want round robin queuing of HTTP sessions enabled among servants.

Data Type	Boolean
Default	true
Used by Daemon	No

Certificate mapping file entries

The following is the syntax for entries in a certificate mapping file.

```
SSLServerCert label ipaddress
```

where:

label Is the label of the server certificate in single or double quotes. If the label itself contains a single quote, double quotes are required as the delimiter.

ipaddress

Is the IP address of the server from which the request was received.

Examples:

```
SSLServerCert 'My Certificate Label' 9.57.4.29
```

```
SSLServerCert "My Co.'s Certificate" 9.57.4.30
```

Managing shared libraries

Shared libraries are files used by multiple applications. Each shared library consists of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries. You can use shared libraries to reduce the number of duplicate library files on your system.

Before you begin

Your applications use the same library files. The applications already are deployed on a server or you currently are deploying the applications.

About this task

Suppose that you have four applications that use the same library file, `my_sample.jar`. Instead of having four copies of `my_sample.jar` on your system after the four applications are deployed, you can define a shared library for `my_sample.jar` and have the four deployed applications use that one `my_sample.jar` library file.

Isolated shared libraries provide another way to reduce the number of library files. Isolated shared libraries each have their own class loader, enabling a single instance of the classes to be shared across the applications. Each application can specify which isolated shared libraries that it wants to reference. Different applications can reference different versions of the isolated shared library, resulting in a set of applications sharing an isolated shared library. With isolated shared libraries, some applications can share a single copy of Library A, Version 1 while other applications share a single copy of Library A, Version 2, for a total of two instances in memory.

Using the administrative console, you can define shared libraries for the library files that multiple applications use and then associate the libraries with specific applications or modules or with an application server. Guidelines for associating shared libraries are as follows:

- Associate a shared library file with an application or module to load the classes represented by the shared library in a local class loader, which can be an application-wide or module-wide class loader.
- Associate an isolated shared library file with an application or module to load the classes represented by the shared library in a separate class loader created for that shared library.
- Associate a shared library file with a server to load the classes represented by the shared library in a server-wide class loader. This class loader is the parent of the application class loader, and the WebSphere Application Server extensions class loader is its parent. Associating a shared library file with a server associates the file with all applications on the server.
- Do not associate an isolated shared library file with a server if you want a separate class loader for a shared library. If you associate the shared library with a server, the product ignores the isolation setting and still adds files in the shared library to the application server class loader. That is, associating an isolated shared library file with a server associates the file with all applications on the server. The product does not use an isolated shared library when you associate the shared library with a server. Associate an isolated shared library with an application or module.

Instead of using the administrative console to associate a shared library with an application, you can use an installed optional package. You associate a shared library to an application by declaring the dependent library `.jar` file in the `MANIFEST.MF` file of the application. Refer to the Java 2 Platform, Enterprise Edition (J2EE) 1.4 specification, section 8.2 for an example.

- Use the administrative console to define a shared library.
 1. Create a shared library.

On a single-server product, you can define a shared library at the cell, node, or server level.

On a multiple-server product, you can define a shared library at the cell, node, server, or cluster level.

Defining a library at one of these levels does not automatically place the library into a class loader. You must associate the library with an application, module, or server before the product loads the classes represented by the shared library into a local or server-wide class loader.

2. Associate each shared library with an application, module, or server.
 - Associate a shared library with an application or module that uses the shared library file.

If you enabled the **Use an isolated class loader for this shared library** setting when creating the shared library, associate the isolated shared library with an application or module to use a separate class loader for the shared library.
 - Associate a shared library with an application server so every application on the server can use the shared library file.
- Use an installed optional package to declare a shared library for an application.
- Remove a shared library.
 1. Click **Environment** → **Shared libraries** in the console navigation tree to access the Shared libraries page.
 2. Select the library to be removed.
 3. Click **Delete**.

The list of shared libraries is refreshed. The library file no longer displays in the list.

Creating shared libraries

Shared libraries are files used by multiple applications. Create a shared library to reduce the number of duplicate library files on your system.

Before you begin

Determine the full path name or directory of each library file for which you want a shared library.

About this task

To make a library file available to multiple applications deployed on a server, create one or more shared libraries for library files that your applications need. When you create the shared libraries, you can use variables within the library file class paths.

You can create one shared library that points to multiple files or directories. This enables you to maintain a single shared library for files that your applications need.

Or you can create a shared library for each library file that your applications need. This approach is recommended only when you have few library files and few applications that use the files. After you create a shared library, you associate it with each application that uses the library files. If you have multiple shared libraries and multiple applications that use the library files, you must complete many steps to create and associate those shared libraries. It is simpler to use one shared library for related files.

Use the Shared libraries page to create and configure shared libraries.

1. Go to the Shared libraries page.

Click **Environment** → **Shared libraries** in the console navigation tree.
2. Select a shared library scope.

Change the scope of the collection table to see what shared libraries are in a particular cell, node or server.

 - a. Select a cell, node, or server.

On a multiple-server product, you also can select a cluster. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** → **Clusters** → **WebSphere application server clusters**).

b. Click **Apply**.

After creating a shared library, you can see whether a shared library can be used on a specific node. Select a scope to see what shared libraries are available to applications installed on or mapped to that scope.

3. Click **New**.

4. Configure the shared library.

a. On the shared library settings page, specify the name, class path, and any other variables for the library file that are needed.

If the shared library specifies a native library path, refer to “Configuring native libraries in shared libraries.”

To have only one instance of a version of a class shared among applications or modules, make the shared library an isolated shared library. Select **Use an isolated class loader for this shared library**. Using an isolated shared library can reduce the memory footprint when a large number of applications share the library.

b. Click **Apply**.

What to do next

Using the administrative console, associate your shared libraries with specific applications or modules or with the class loader of an application server. Associating a shared library file with a server class loader associates the file with all applications on the server.

If you enabled the **Use an isolated class loader for this shared library** setting when creating your shared library, associate the shared library with applications or Web modules. If you associate the shared library with a server, the product ignores this setting and still adds files in the shared library to the application server class loader. The product does not use an isolated shared library when you associate the shared library with a server.

Alternatively, you can use an installed optional package to associate your shared libraries with an application.

Configuring native libraries in shared libraries

Native libraries are platform-specific library files, including .dll, .so, or *SRVPGM objects, that can be configured within shared libraries. Native libraries are visible to an application class loader whenever the shared library is associated with an application. Similarly, native libraries are visible to an application server class loader whenever the shared library is associated with an application server.

Before you begin

When designing a shared library, consider the following conditions regarding Java native library support:

- The Java virtual machine (JVM) allows only one class loader to load a particular native library.
- There is no application programming interface (API) to unload a native library from a class loader.

Native libraries are unloaded by the JVM when the class loader that found the library is collected from the heap during garbage collection.

- Application server class loaders, unlike the native JVM class loader, only load native shared libraries that use the default operating system extension for the current platform. For example, on AIX, native shared libraries must end in .a when loaded by application server class loaders. The JVM class loader loads files ending in .a or .so.
- Application server class loaders persist for the duration of the application server.

- Application class loaders persist until an application is stopped or dynamically reloaded.
If a shared library that is configured with a native library path is associated with an application, whenever the application is restarted or dynamically reloaded the application might fail with an `UnsatisfiedLinkError` indicating that the library is already loaded. The error occurs because, when the application restarts, it invokes the shared library class to reload the native library. The native library, however, is still loaded in memory because the application class loader which previously loaded the native library has not yet been garbage collected.

- Only the JVM class loader can load a dependent native library.

For example, if *NativeLib1* is dependent on *NativeLib2*, then *NativeLib2* must be visible to the JVM class loader. The path containing *NativeLib2* must be specified on Java library path defined by the `LIBPATH` environment variable.

The `LIBPATH` (`java.library.path`) property is configured using the `process_name_region_libpath` environment variable, such as `control_region_libpath`, `server_region_libpath`, or `adjunct_region_libpath`. For instructions on how to set the region `LIBPATH` variables, see “Changing the values of variables referenced in BBOM00011 messages” on page 175.

If a native library configured in a shared library is dependent on other native libraries, the dependent libraries must be configured on the `LIBPATH` of the JVM hosting the application server in order for that library to load successfully.

About this task

When configuring a shared library on a shared library settings page, if you specify a value for **Native library path**, the native libraries on this path are not located by the WebSphere Application Server application or shared library class loaders unless the class which loads the native library was itself loaded by the same class loader.

Because a native library cannot be loaded more than once by a class loader, it is preferable for native libraries to be loaded within shared libraries associated with the class loader of an application server, because these class loaders persist for the lifetime of the server.

1. Implement a static method in the class that loads the native library.

In the class that loads the native library, call `System.loadLibrary(native_library)` in a static block. For example:

```
static {System.loadLibrary("native_library");
```

native_library loads during the static initialization of the class, which occurs exactly once when the class loads.

2. On the shared library settings page, set values for **Classpath** and **Native library path** that enable the shared library to load the native library.

If you want to associate your shared library with an application or module, also select **Use an isolated class loader for this shared library**. If you do not enable this setting, associate the shared library with an application server.

3. Associate the shared library.

- If you did not enable **Use an isolated class loader for this shared library**, associate the shared library with an application server.

Associating a shared library with the class loader of an application server, rather than with an application, ensures that the shared library is loaded exactly once by the application server class loader, even though applications on the server are restarted or dynamically reloaded. Because the native library is loaded within a static block, the native library is never loaded more than once.

- If you enabled **Use an isolated class loader for this shared library**, associate the shared library with an application or module.

Associating an isolated shared library file with an application or module loads the classes represented by the shared library in a separate class loader created for that shared library. Do not associate an isolated shared library file with a server if you want a separate class loader for a

shared library. If you associate the shared library with a server, the product ignores the isolation setting and still adds files in the shared library to the application server class loader. That is, associating an isolated shared library file with a server associates the file with all applications on the server.

The class loader created for an isolated shared library does not reload and, like a server class loader, exists for the lifetime of a server. For shared native libraries, you can use an isolated shared library to avoid errors resulting from reloading of native libraries.

What to do next

To verify that an application can use a shared library, test the application or examine the class loader in the Class loader viewer. Click **Troubleshooting** → **Class loader viewer** → *module_name* → **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Shared library collection

Use this page to define a list of shared library files that deployed applications can use.

To view this administrative console page, click **Environment** → **Shared libraries**.

Change the scope to see what shared libraries are in a particular node or server. By default, a shared library is accessible to applications deployed (or installed) on the same node as the shared library file. To change the scope, select the cell, a node, or a server under **Scope**.

On a multiple-server product, you also can select a cluster. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** → **Clusters** → **WebSphere application server clusters**). The cluster scope limits the scope of a shared library to cluster members of a particular cluster.

Select a scope before you click **New** and create a shared library. After you create a shared library and map an application to the selected scope, you can associate the shared library with the application or its modules.

- To associate a shared library with an application or module, use the Shared library references page for the application. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name* → **Shared library references**.
- To associate a shared library with a server class loader, use the settings page for the library reference for the server class loader. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name* → **Java and Process Management** → **Class loader** → *class_loader_ID* → **Shared library references** → *shared_library_name*.

Name

Specifies a name for the shared library.

Description

Describes the shared library file.

Shared library settings

Use this page to make a library file available to deployed applications.

To view this administrative console page, click **Environment** → **Shared libraries** → *shared_library_name*.

Scope:

Specifies the level of the location of the shared library configuration file.

On single-server installations, the shared library has its configuration file in a location that pertains to the cell, node, or server level.

On multiple-server installations, the shared library has its configuration file in a location that pertains to the cell, node, server, or cluster level.

Data type String

Name:

Specifies a name for the shared library.

Data type String

Description:

Describes the shared library.

Data type String

Classpath:

Specifies a list of paths that the product searches for classes and resources of the shared library.

If a path in the list is a file, the product searches the contents of that Java archive (JAR) or compressed (zip) file. If a path in the list is a directory, then the product searches the contents of JAR and zip files in that directory. For performance reasons, the product searches the directory itself only if the directory contains subdirectories or files other than JAR or zip files.

Press Enter to separate class path entries. Entries must not contain path separator characters such as a semicolon (;) or colon (:). Class paths can contain variable names that can be substituted using a variable map.

Data type String

Units Class path

Native library path:

Specifies the class path for locating platform-specific library files for shared library support; for example, .dll, .so, or *SRVPGM objects.

If you specify a value for **Native library path**, the native libraries are not located by application or shared library class loaders unless the following conditions exist:

- A class loads the native libraries.
- The application invokes a method in this class which loads the libraries.

For example, in the class that loads the native library, call `System.loadLibrary(native_library)` in a static block:

```
static {System.loadLibrary("native_library");}
```

- The **Classpath** specified on this page contains the class that loads the libraries.

Native libraries cannot be loaded more than once by a class loader. Thus, it is preferable for native libraries to use an isolated shared library or to be loaded within shared libraries associated with the class loader of an application server. See the **Use an isolated class loader for this shared library** setting.

Data type String

Units Class path

Use an isolated class loader for this shared library:

Specifies whether the shared library has a single isolated shared library shared across its associated applications or Web modules.

Note: An isolated shared library enables one instance of the library classes to be shared only among associated applications and Web modules. An isolated shared library enables multiple applications or Web modules to share a common set of classes across a subset of the applications. Further, an isolated shared library supports versioning and loads the minimum number of library copies. The class loader created for an isolated shared library does not reload and, like a server class loader, exists for the lifetime of a server. For shared native libraries, you can use an isolated shared library to avoid errors resulting from reloading of native libraries.

The default, `false`, is not to isolate the shared library so that each application loads its own instances of the shared library classes.

Using an isolated shared library can reduce the memory footprint when a large number of applications share the library. If you select this option, associate the shared library with applications or Web modules.

Note: If you associate the shared library with a server, the product ignores this setting and still adds files in the shared library to the application server class loader. The product does not use an isolated shared library when you associate the shared library with a server. To use an isolated shared library, you must associate the shared library with applications or Web modules.

Selecting this option affects the class loader order of the associated application or Web module. If the class loader order for a class loader associated with an isolated shared library is **Classes loaded with the parent class loader first** (Parent first), the class loader checks whether a class can be loaded in the following order:

1. Checks whether the associated library class loaders can load the class.
2. Checks whether its parent class loader can load the class.
3. Checks whether it (application or WAR module class loader) can load the class.

If the order is **Classes loaded with the local class loader first (Parent last)**, the class loader checks in the following order:

1. Checks whether it (application or WAR module class loader) can load the class.
2. Checks whether the associated library class loaders can load the class.
3. Checks whether its parent class loader can load the class.

This setting maps to the `isolatedClassLoader` Boolean attribute of the Library object.

Boolean `false`

Associating shared libraries with applications or modules

You can associate a shared library with an application or module. Classes represented by the shared library are then loaded in the application's class loader, making the classes available to the application.

Before you begin

This topic assumes that you have defined a shared library. The shared library represents a library file used by multiple deployed applications.

You can define a shared library at the cell, node, server, or cluster level.

On a multiple-server product, you also can define a shared library at the cluster level. To see the cluster scope, you first must create a cluster on the Server clusters page (**Servers** → **Clusters** → **WebSphere application server clusters**).

This topic also assumes that you want to use the administrative console, and not an installed optional package, to associate a shared library with an application.

About this task

To associate a shared library with an application or module, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with an application, do not associate the same shared library with a server class loader.

1. If you have not done so already, map your application to a target server that is within the scope of the shared library.
For example, if the shared library scope is the *my_cluster* cluster, map your application to the target *my_cluster* cluster.
2. Click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name* → **Shared library references** in the console navigation tree to access the Shared library references page.
3. On the Shared library references page, select an application or module to which you want to associate a shared library.
4. Click **Reference shared libraries**.
5. On the Shared library mapping page, select one or more shared libraries that the application or modules use in the **Available** list, click >> to add them to the **Selected** list, and click **OK**.
6. Repeat steps 2 through 4 until you define a library reference instance for each shared library that your application or module requires.
7. On the Shared library references page, click **OK**.
8. Save the changes to the configuration.

Results

When you run the application, classes represented by the shared library are loaded in the application class loader.

The classes are now available to the application or module.

What to do next

To verify an association between an application and a shared library, examine the application class loader in the Class loader viewer. Click **Troubleshooting** → **Class loader viewer** → *module_name* → **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Shared library reference and mapping settings

Use the Shared library references and Shared library mapping pages to associate defined shared libraries with an application or Web module. A shared library is an external Java archive (JAR) file that is used by one or more applications. Using shared libraries enables multiple applications deployed on a server to use a single library, rather than use multiple copies of the same library. After you associate shared libraries with an application or module, the application or module class loader loads classes represented by the shared libraries and makes those classes available to the application or module.

To view the Shared library references console page, click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name* → **Shared library references**. To view the

Shared library mapping page, click **Reference shared libraries** on the Shared library references page. These pages are the same as the Map shared libraries and Map shared libraries to an entire application or module pages in the application installation and update wizards.

On the Shared library references page, the first element listed is the application. The other elements are modules in the application.

To associate shared libraries with your application or module:

1. Select an application or module.
2. Click **Reference shared libraries**.
3. On the Shared library mapping page, select one or more shared libraries that the application or modules uses in the **Available** list, click >> to add them to the **Selected** list, and click **OK**.

A defined shared library for a file that your application or module uses must exist to associate your application or module to the library.

If no shared libraries are defined and the application is installed already, on the Shared library mapping page, click **New** and define a shared library.

You can otherwise define a shared library as follows:

1. Click **Environment** → **Shared libraries**.
2. Specify whether the shared library is visible at the cell, node or server level.
3. Click **New**.
4. On the settings page for the new shared library, specify a name and one or more class paths. If the libraries are platform-specific files such as .dll, .so, or *SRVPGM objects, also specify a native library path. Then, click **Apply**.
5. Save the administrative configuration.

Application:

Specifies the name of the application that you are installing or that you selected on the Enterprise applications page.

Module:

Specifies the name of the module associated with the shared libraries.

URI:

Specifies the location of the module relative to the root of the application EAR file.

Shared libraries:

Specifies the name of the shared library files associated with the application or module.

Associating shared libraries with servers

You can associate shared libraries with the class loader of a server. Classes represented by the shared library are then loaded in a server-wide class loader, making the classes available to all applications deployed on the server.

Before you begin

This topic assumes that you have defined a shared library. The shared library represents a library file used by multiple deployed applications.

About this task

To associate a shared library with the class loader of a server, create and configure a library reference using the administrative console. A library reference specifies the name of the shared library file.

If you associate a shared library with a server class loader, do not associate the same shared library with an application.

1. Configure class loaders for applications deployed on the server.
 - a. Click **Servers** → **Server Types** → **WebSphere application servers** → *server_name* to access the application server setting page.
 - b. Set values for the application **Class loader policy** and **Class loading mode** of the server.
For information on these settings, see Application server settings in the *Administering applications and their environment* PDF.
2. Create a library reference for each shared library file that your application needs.
 - a. In the administrative console, click **Servers** → **Server Types** → **WebSphere application servers** → *server_name* → **Java and Process Management** → **Class loader** → *class_loader_ID*.
 - b. Click **Shared library references** to access the Library reference page.
 - c. Click **Add**.
 - d. On the library reference settings page, name the library reference. The name identifies the shared library file that your application uses.
 - e. Click **Apply**. The name of the library reference is shown in the list on the Library reference page.

Repeat the previous steps until you define a library reference for each shared library that your application needs.

What to do next

To verify that an application can use a shared library, test the application or examine the class loader in the Class loader viewer. Click **Troubleshooting** → **Class loader viewer** → *module_name* → **Table View**. The classpath of the application module class loader lists the classes used by the shared library.

Installed optional packages

Installed optional packages enable applications to use the classes in Java archive (.jar) files without having to include them explicitly in a class path. An installed optional package is a .jar file containing specialized tags in its manifest file that enable the application server to identify it. An installed optional package declares one or more shared library .jar files in the manifest file of an application. When the application is installed on a server or cluster, the classes represented by the shared libraries are loaded in the class loader of the application, making the classes available to the application.

When a Java Platform, Enterprise Edition (Java EE) application is installed on a server or cluster, dependency information is specified in its manifest file. The product reads the dependency information of the application (.ear file) to automatically associate the application with an installed optional package .jar file. The product adds the .jar files in associated optional packages to the application class path. Classes in the installed optional packages are then available to application classes.

Installed optional packages used by the product are described in section 8.2 of the Java 2 Platform, Enterprise Edition (J2EE) specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

The product supports using the manifest file (manifest.mf) in shared library .jar files and application .ear files. The product does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (<http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html>), which primarily serve the applet environment. The product ignores applet-specific tags within manifest files.

Sample manifest.mf file

A sample manifest file follows for an application `app1.ear` that refers to a single shared library file `util.jar`:

```
app1.ear:
  META-INF/application.xml
  ejb1.jar:
    META-INF/MANIFEST.MF:
      Extension-List: util
      util-Extension-Name: com/example/util
      util-Specification-Version: 1.4
    META-INF/ejb-jar.xml

util.jar:
  META-INF/MANIFEST.MF:
    Extension-Name: com/example/util
    Specification-Title: example.com's util package
    Specification-Version: 1.4
    Specification-Vendor: example.com
    Implementation-Version: build96
```

The syntax of a manifest entry depends on whether the entry applies to a member with a defining role (the shared library) or a member with a referencing role (a Java EE application or a module within a Java EE application).

Manifest entry tagging

Main tags used for manifest entries include the following:

Extension-List

A required tag with variable syntax. Within the context of the referencing role (application's manifest), this is a space delimited list that identifies and constructs unique `Extension-Name`, `Extension-Specification` tags for each element in the list. Within the context of the defining role (shared library), this tag is not valid.

Extension-Name

A required tag that provides a name and links the defining and referencing members. The syntax of the element within the referencing role is to prefix the element with the `<ListElement>` string. For each element in the `Extension-List`, there is a corresponding `<ListElement>-Extension-Name` tag. The defining string literal value for this tag (in the above sample `com/example/util`) is used to match (in an equality test) the corresponding tags between the defining and referencing roles.

Specification-Version

A required tag that identifies the specification version and links the defining and referencing members.

Implementation-Version

An optional tag that identifies the implementation version and links the defining and referencing members.

Further information on these tags is in the `.jar` file specification at <http://java.sun.com/j2se/1.4.2/docs/guide/jar/jar.html#Manifest%20Specification>.

Using installed optional packages

You can associate one or more shared libraries with an application using an installed optional package that declares the shared libraries in the application's manifest file. Classes represented by the shared libraries are then loaded in the application's class loader, making the classes available to the application.

Before you begin

Read about installed optional packages in “Installed optional packages” on page 140 and in section 8.2 of the Java 2 Platform, Enterprise Edition (J2EE) specification, Version 1.4 at http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf.

WebSphere Application Server does not support the Java 2 Platform Standard Edition (J2SE) Installed Optional Package semantics used in the J2SE specification (<http://java.sun.com/j2se/1.3/docs/guide/extensions/spec.html>), which primarily serve the applet environment. WebSphere Application Server ignores applet-specific tags within manifest files.

About this task

Installed optional packages expand the existing shared library capabilities of an application server. Prior to Version 6.0, an administrator was required to associate a shared library to an application or server. Installed optional packages enable an administrator to declare a dependency in an application’s manifest file to a shared library, with installed optional package elements listed in the manifest file, and automatically associate the application to the shared library. During application installation, the shared library .jar file is added to the class path of the application class loader.

If you use an installed optional package to associate a shared library with an application, do not associate the same shared library with an application class loader or a server class loader using the administrative console.

1. Assemble the library file, including the manifest information that identifies it as an extension. Two sample manifest files follow. The first sample manifest file has application `app1.ear` refer to a single shared library file `util.jar`:

app1.ear:

```
META-INF/application.xml
ejb1.jar:
  META-INF/MANIFEST.MF:
    Extension-List: util
    util-Extension-Name: com/example/util
    util-Specification-Version: 1.4
  META-INF/ejb-jar.xml
```

util.jar:

```
META-INF/MANIFEST.MF:
  Extension-Name: com/example/util
  Specification-Title: example.com's util package
  Specification-Version: 1.4
  Specification-Vendor: example.com
  Implementation-Version: build96
```

The second sample manifest file has application `app1.ear` refer to multiple shared library .jar files:

app1.ear:

```
META-INF/application.xml
ejb1.jar:
  META-INF/MANIFEST.MF:
    Extension-List: util1 util2 util3
    Util1-Extension-Name: com/example/util1
    Util1-Specification-Version: 1.4
    Util2-Extension-Name: com/example/util2
    Util2-Specification-Version: 1.4
    Util3-Extension-Name: com/example/util3
    Util3-Specification-Version: 1.4
  META-INF/ejb-jar.xml
```

util1.jar:

```
META-INF/MANIFEST.MF:
  Extension-Name: com/example/util1
  Specification-Title: example.com's util package
```


Specification-Version: 1.4
Specification-Vendor: example.com
Implementation-Version: build96

util2.jar:

META-INF/MANIFEST.MF:
Extension-Name: com/example/util2
Specification-Title: example.com's util package
Specification-Version: 1.4
Specification-Vendor: example.com
Implementation-Version: build96

util3.jar:

META-INF/MANIFEST.MF:
Extension-Name: com/example/util3
Specification-Title: example.com's util package
Specification-Version: 1.4
Specification-Vendor: example.com
Implementation-Version: build96

2. Create a shared library that represents the library file assembled in step 1. This installs the library file as a shared library.
3. Copy the shared library .jar file to the cluster members.
4. Assemble the application, declaring in the application manifest file dependencies to the library files named the manifest created for step 1.
See the *Developing and deploying applications* PDF for more information.
5. Install the application on the server or cluster.
See the *Developing and deploying applications* PDF for more information.

Results

During application installation, the shared library .jar files are added to the class path of the application class loader.

Library reference collection

Use this page to view and manage library references that define how to use global libraries. For example, you can use this page to associate shared library files with a deployed application.

To view this administrative console page, click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and Process Management** → **Class loader** → **class_loader_ID** → **Shared library references** .

If no shared libraries are defined in your environment, such as at the node or server scope, after you click **Add** a message is displayed stating that you must define a shared library before you can create a library reference. A shared library is a container-wide library file that deployed applications can use. To define a shared library, click **Environment** → **Shared libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library. After you define a shared library, return to this page, click **Add**, and create a library reference.

Library name

Specifies a name for the library reference.

Library reference settings

Use this page to define library references, which specify how to use global libraries.

To view this administrative console page, click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Java and Process Management** → **Class loader** → **class_loader_ID** → **Shared library references** → **library_reference_name**.

A shared library is a container-wide library file that deployed applications can use. To define a shared library, click **Environment** → **Shared libraries** and specify the scope of the container. Then, click **New** and specify a name and one or more paths for the shared library.

Library name:

Specifies the name of the shared library to use for the library reference.

Data type String

Application server naming conventions

Before you install a new WebSphere Application Server for z/OS environment, it is important to carefully plan your naming convention. Your naming convention should be able to grow with your system when you increase the number of cells, nodes, servers, and clusters. It should also be able to accommodate Sysplex and LPAR names, as well as instances such as test, integration, and production stages in your environment.

Application servers are like IMS™ or CICS® regions.

- They contain tailored procedures for the controllers and servants.
- They contain tailored environmental variables for each instance of a server.
- They use WLM Classification of regions, working within the regions, and are defined as application environments.
- They may be self-contained or dependent on other servers.
- They need RACF definitions for Control and Server STC (user IDs, resource profiles), as well as UNIX permissions.
- Their users must be allowed to access the servers and to use various objects within them.

The product environment consists of a number of address spaces which require the installation to manage security profiles, workload classification constructs, and so on. To create, manage, and recognize application servers, it may be helpful to create a template for naming your servers and server instances. You can find an example template in the *Installing your application serving environment* PDF.

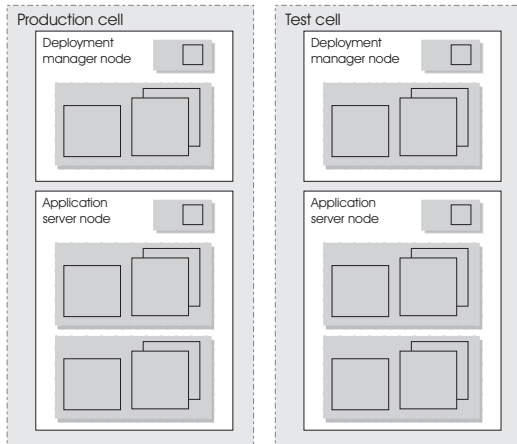
It is also important to plan the naming conventions for your data sets carefully.

- SMP/E target data sets, depending on your maintenance process (regular data sets and the HFS, including its mount points)
- Customization HFS, including its mount point
- HLQ for your customization data sets (*.CNTL, *.DATA, and *.SAVDCFG)
- Error logstream names
- DB2 collection and package names

Test cells and production cells

If you require complete availability of your production system, this configuration eliminates the risk of including production and test in the same cell.

As the graphic below indicates, placing test and production servers into separate cells eliminates all local sharing between test and production and provides the highest risk reduction possible.



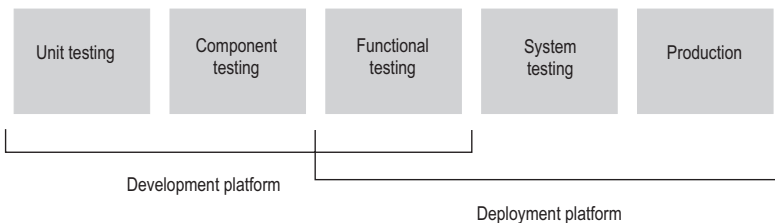
Testing and production phases

Before explaining the test and production configurations for the product, you must understand which test phase should be done on the z/OS platform and which should be done on other platforms.

Note: Sharing resources between a production workload and a test workload can expose the production workload to a set of error conditions to which it is not exposed if the production and test workloads run in different cells. If possible, you should run production and test workloads in separate cells on your system.

Before setting up your test and production configurations for the product, you must understand which test phases should be done on the z/OS platform and which should be done on other platforms. The following sections explain the different phases:

- Unit test phase
- Component test phase
- Function test phase
- System test phase
- Production phase



Unit test phase

Applications that you plan on running in a z/OS environment should be developed on a distributed operating system, such as Windows or Linux Intel®, on which the product is installed. These development environments contain assembly tools for Web content delivery that are not available on z/OS. The IBM tooling solution assumes that you develop enterprise beans in one of these tools and perform basic testing of the business logic in the distributed environment before moving the application to the z/OS environment.

Component test phase

Component testing involves the joining together of several enterprise beans into logical components, providing them with access to data, and testing them together. While this can be done on a z/OS platform, it is recommended that you do this level of testing on a distributed platform. Performing this type of testing on a distributed platform enables a small team of developers to join the code pieces together and test the interactions. This type of testing focuses on the individual beans and their relationships to each other rather than z/OS platform functions and features.

Function test phase

Function testing involves joining the various components together, connecting them to test data in the target database, and validating the function that the application provides. Where this test is performed depends on the function and its data requirements. If the target deployment platform is z/OS, you might want to do this level of testing on z/OS. In this situation you should install the applications that you are testing on one or more servers that are only used for testing.

When you install the application on a test server, define where in the JNDI directory the references to the application are stored, and then configure the test clients such that they know the location of the test application. The test clients can then drive requests against the test server to perform the functional testing. You can use remote debugging tools to diagnose problems you encounter along the way.

System test phase

Before you put an application into production on z/OS, you should install the application into a system test environment on z/O and simulate a real workload on that application. When setting up your system test environment, you should define an additional test server on a cell that is dedicated to the test system, and install the application onto that server. When installed, enterprise beans that are part of the application should be registered in a different subtree of the JNDI directory. This normally happens by default but it is good to verify that this registration occurs. The test clients must be configured to the version of the application that is being tested before you run your tests.

Production phase

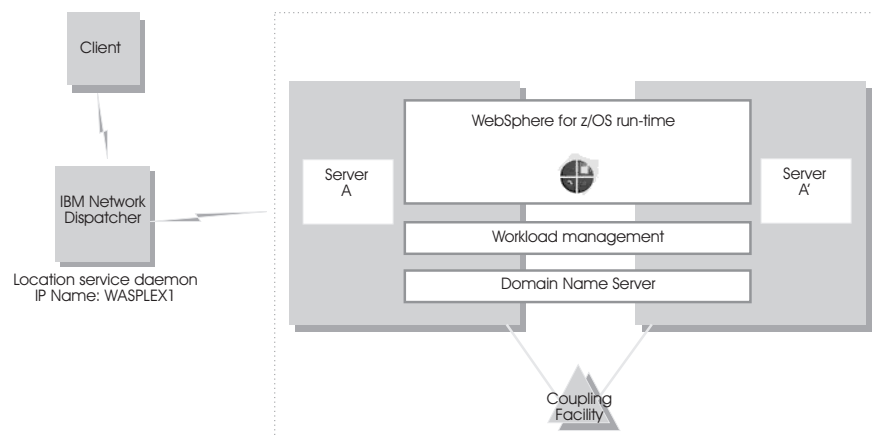
After you are satisfied with the functional and system testing, install the application in a cell that is used for production . The difference between a production cell and a test cell is whether the remote debugger is allowed to be attached. Normally, it is not acceptable for a production workload to stop because a remote debugging request is sent to the cell.

Load Balancer

The load balancer is a router that handles network requests for the cell. This component was previously called the network dispatcher,

Characteristics of such a configuration are:

- The location service daemon IP Name is associated with the IP address of the router.
- Load Balancer cooperates with workload management to route requests through the cell. The client never sees a change in IP addresses.
- The implication for clients is that they can cache the IP addresses, because this configuration does not change them dynamically.



Setting up peer restart and recovery

To allow the product to restart on an alternate system, the following prerequisites must be installed on every system (your original system as well as any systems intended for recovery) before reconfiguring the ARM policies to enable peer restart and recovery.

Before you begin

Note: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See the topic *Transaction support in WebSphere Application Server* for more information about the integrated high availability support for the transaction service subcomponent and how to configure it for peer recovery of transactions being processed on a application server that fails.

You must also make sure all of the systems, where you might need to perform restart, are part of the same RRS log group.

- z/OS Version 1.2 or higher
- BCP APAR OA01584
- RRS APARs OA02556 and OA2556
- WebSphere Application Server Version 5 or higher

Installing the prerequisite service updates on all of these systems will not hinder your current running environment if you want to continue to only restart in place. However, if this service is not installed, there is a possibility that the controller will not be able to move back. OTS will attempt to restart on the alternate system and fail. If there are any URs that are unresolved with RRS once this happens, the controller will not be allowed to restart on the home system until RRS is cancelled on the alternate system. For more information on OTS and RRS, see *z/OS MVS Programming: Resource Recovery*.

If you do not plan to use peer restart and recovery, you do not need to abide by these functional prerequisites. Your system will instead use the restart-in-place function.

The following products all support RRS. Individually, they also support peer restart and recovery, providing the above prerequisites are all properly installed:

- DB2 Version 7 or higher
- IMS Version 8 or higher
- CICS Version 1.3 or higher
- MQSeries® Version 5.2 or higher

In addition to the preceding products, many JTA XAResource Managers can be used to assist in a the product peer restart and recovery. Consult your JTA XAResource Manager's documentation to determine if it supports restarting on an alternate system.

Note: When setting up the ARM policy for a sysplex, make sure that both systems have the same level of the Application Server installed. For example, you cannot use an application server that is running WebSphere Application Server Version 5.1 to perform peer restart and recovery for an application server that is running WebSphere Application Server Version 6.0.1.

Prior to using peer restart and recovery:

- You must ensure that the location service Daemon and node agent are already running on all systems that might be used for recovery. Otherwise, the recovering system might attempt to recover on a system that is not running the location service Daemon and node agent. If this happens, the server will fail to start, and recovery will fail.

Clients will see a performance impact if the systems are running at capacity. In an attempt to minimize the memory and CPU impact on the alternate system, the enterprise bean and Web containers are not restarted for servers running in peer-restart mode. This means that application servers that are in the state of being recovered will not be able to accept any inbound work.

About this task

After the prerequisites are installed, starting a server on a system to which it was not configured implicitly places the server into peer restart and recovery mode. If you configured your XA Partner log to write to a non-shared HFS, or if you are using a JTA XA Resource Manager, you need to perform the following steps before starting a server:

1. (Required only if you are using a non-shared HFS.) Enable non-shared HFS support. When using a non-shared HFS, the configuration settings must be replicated across the different systems in the sysplex. This is done automatically by the deployment manager and node agent. To enable this support, each node agent in your configuration must be set as a recovery node. This change is made in the administrative console:
 - a. In the administrative console navigation, select **System Administration > Node agents**.
 - b. Select a node agent from the list.
 - c. In the Additional Properties section, select **File Synchronization Service**.
 - d. In the Additional Properties section, , select **Custom properties**.
 - e. Select **New**.
 - f. Enter recoveryNode for **Name**, and true for **Value**. The **Description** field can be left blank.
 - g. Repeat steps 3-7 for each node agent in your configuration.
 - h. Save your configuration.
2. (Required only if you are using JTA XAResource Managers.) Make appropriate logs and classes are available on the alternate system If you plan to use peer restart and recovery, and your applications access JTA XAResource Managers, you must ensure that the appropriate logs and classes are available on the alternate system.
 - a. Point the product variable `TRANLOG_ROOT` to a shared HFS. The `TRANLOG_ROOT` variable must point to a shared HFS, to which all systems in the cell can write. The XA partner log is stored here, and the alternate system must be able to read and update this log.

Use the administrative console to set the product variable, `TRANLOG_ROOT`, to the directory of a shared HFS, to which all systems in the cell can write.

In the administrative console, click **Environment > Manage WebSphere Variables**. Then click on the **TRANLOG_ROOT** variable to bring up an new window in which you can specify the directory of the shared HFS.

- b. Store the driver (i.e., JDBC Driver, JMS Provider, or JCA Resource Adapter, etc.) for each JTA XAResource Manager in an HFS that is readable by all systems in the cell. For example, if your connector is a JDBC driver for a database, the driver would likely be stored in a read-only HFS that is accessible by all systems in the sysplex. This allows the alternate system to read the saved classpath for the resource, and reconstruct it during a restart.

If the connector used to access a JTA XAResource Manager is not stored in an HFS that is readable by all systems that might be used for recovery, when an application server restarts on an alternate system, it will either appear that there is no XA recovery work to do, or it will be impossible to load the classes necessary to communicate with the JTA XAResource Manager

3. Resolve InDoubt units.

During a recovery, there will be instances when manual intervention is required to resolve InDoubt units. You will need to use RRS panels for this manual intervention.

Peer restart and recovery

The goal of every system is to have as little downtime as possible. Sometimes, however, system failures are inevitable. For example, a system failure might occur because the power unexpectedly goes out in your main system. When a system failure occurs, a restart action you can take is to restart on a peer system in the sysplex. This type of restart uses the peer restart and recovery function. Starting a server on a system to which it was not configured implicitly places it into peer restart and recovery mode.

Note: Peer restart and recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery.

When you experience a main system failure that results in InDoubt transactions with unknown outcomes, you need to obtain those intended transactional outcomes (ideally correctly) before the data can be utilized again. Peer restart and recovery provides an automated means of accomplishing this by restarting the controller on a peer system so that the "locks" that block the data can be dropped and the outcomes determined. This is in contrast to how a system usually handles a failure by automatically rolling back.

If a failure occurs, automatic restart management:

- Can restart the product and related servers on the same system, or
- Can use the peer restart and recovery function to restart related servers on an alternate system in the cell.

The server is not a recoverable *resource* manager. It is a recoverable *communication manager*. It has no recoverable locks of its own and it does not need to manage locks nor manage lock states in a log. It just needs to make sure that both callers and callees are connected in each of the communications sessions of a distributed transaction.

Peer restart and recovery restarts the controller on another system and goes through the transaction restart and recovery process so that we can assign outcomes to transactions that were in progress at the time of failure. During this transaction restart and recovery process, data might be temporarily inaccessible until the recovery process is complete. The restart and recovery process does not result in lost data.

Resource managers, such as DB2, that were being accessed at the time of failure may hold locks that are scoped to a transaction UR (unit of recovery). Once an outcome has been assigned to a UR, the resource managers will, generally, drop those locks.

When might PRR fail to recover servers

The major reason for peer restart and recovery (PRR) failure is if you experience a network outage while in the process of recovering. If the system cannot reach the superior or subordinate because the network is dead, communications cannot reestablish and the transaction cannot completely resolve.

Note: Peer restart and recovery functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of peer restart and recovery for transaction recovery.

When the product cannot automatically resolve all of the URs returned from RRS at restart, RRS will not allow the application server to move back to the home (original) system. If the application server tries to go back while URs are still incomplete, you will receive an error code (C9C2186A) and a message describing an F02 return code from ATRIBRS. In order to get around this, manual resolution is required to mark the server for "restart anywhere." RRS will do that once all of the URs in which the product is involved are *forgotten*. If RRS fails to mark the server *restart anywhere*, the server, upon failure, is required to start on the recovery system. This is not good because it doesn't allow you to move the server back to its true home system.

The ultimate goal of this is to resolve all transactions that the application server (the server instance-owned interests that could not complete recovery) is involved in, and then, if necessary, remove all of the application server interests that remain in those URs. Once that is complete, browsing the RM data log will show if the resource manager is marked "restart anywhere."

You **want** to see:

```
RESOURCE MANAGER=BSS00.SY1.BBOASR4A.IBM  
RESOURCE MANAGER MAY RESTART ON ANY SYSTEM
```

You do **not** want to see:

```
RESOURCE MANAGER=BSS00.SY2.BBOASR4A.IBM  
RESOURCE MANAGER MUST RESTART ON SYSTEM SY2
```

Using RRS panels to resolve InDoubt units of recovery

Use this task to better understand messages received when using peer restart and recovery.

Before you begin

Note: Peer Restart and Recovery (PRR) functionality is deprecated. You should use the integrated high availability support for the transaction service subcomponent, instead of Peer Restart and Recovery for transaction recovery. See the topic *Transaction support in WebSphere Application Server* for more information about the integrated high availability support for the transaction service subcomponent.

There are RRS version requirements that you must heed when using peer restart and recovery. For more information on these requirements, see *z/OS MVS Programming: Resource Recovery*.

About this task

If you receive the console message:

```
BBOT0015D OTS UNABLE TO RESOLVE ALL INCOMPLETE TRANSACTIONS FOR SERVER  
string. REPLY CONTINUE OR TERMINATE.
```

1. Note the server name specified for *string*. in the message.
2. Go to the SYSPRINT, that is the status queue for that server, and search for messages BBOT0019 - BBOT0022 that refer to that server.
3. Read the resulting messages.
4. Stop the server.

RRS does not allow an operator to resolve an InDoubt UR if the DSRM for that UR is active at the time. Therefore, you must stop the server. To do this, reply TERMINATE to the CONTINUE/TERMINATE WTOR.

What to do next

You can use the following non-console messages to trigger restart and recovery automation. These messages provide information about daemon activities. Pay particular attention to the URID, XID FormatId, XID Gtrid, and XIDBqual attributes. You need to use these pieces of information when you manually resolve the relevant units of work via the RRS panels.

- **BBOO003E** WEBSPPHERE FOR z/OS CONTROL REGION string ENDED ABNORMALLY, REASON=*hstring*.
- **BBOO009E** WEBSPPHERE FOR z/OS DAEMON string ENDED ABNORMALLY, REASON= *hstring*.
- **BBOO0171I** WEBSPPHERE FOR z/OS CONTROL REGION string NOT STARTING ON CONFIGURED SYSTEM *string*
- The following messages, which are written only in recovery and restart mode, provide details about transactions that cannot be resolved during restart and recovery:
 - **BBOT0008I** TRANSACTION SERVICE RESTART INITIATED ON SERVER *string*
 - **BBOT0009I** TRANSACTION SERVICE RESTART UR STATUS COUNTS FOR SERVER string: IN-BACKOUT= *dstring*, IN-DOUBT= *dstring*, IN-COMMIT= *dstring*
 - **BBOT0010I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER string IS COMPLETE
 - **BBOT0011I** SERVER *string* IS COLD STARTING WITH RRS
 - **BBOT0012I** SERVER *string* IS WARM STARTING WITH RRS
 - **BBOT0013I** TRANSACTION SERVICE RESTART AND RECOVERY ON SERVER *string* IS COMPLETE. THE SERVER IS STOPPING.
 - **BBOT0014I** TRANSACTION SERVICE RECOVERY PROCESSING FOR RRS URID ' *string* ' IN SERVER *string* IS COMPLETE.
 - **BBOT0016I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS NOT COMPLETE. THE SERVER IS STOPPING DUE TO OPERATOR REPLY.
 - **BBOT0017I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS CONTINUING DUE TO OPERATOR REPLY.
 - **BBOT0018I** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER *string* IS STILL PROCESSING *dstring* INCOMPLETE UNIT(S) OF RECOVERY.
 - **BBOT0019W** UNABLE TO RESOLVE THE OUTCOME OF THE TRANSACTION BRANCH DESCRIBED BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE OTS RECOVERY COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* COULD NOT BE REACHED.
 - **BBOT0020W** UNABLE TO PROVIDE THE SUBORDINATE OTS RESOURCE IN SERVER string ON HOST *string*: *dstring* WITH THE OUTCOME OF THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THIS SERVER HAS BEEN UNABLE TO RESOLVE THE OUTCOME WITH A SUPERIOR NODE.
 - **BBOT0021W** UNABLE TO *string* THE SUBORDINATE OTS RESOURCE IN SERVER *string* ON HOST *string*: *dstring* FOR THE TRANSACTION DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' OR ANOTHER RESOURCE INVOLVED IN THIS UNIT OF RECOVERY BECAUSE ONE OR MORE RESOURCES COULD NOT BE REACHED OR HAVE NOT YET REPLIED.
 - **BBOT0022W** UNABLE TO FORGET THE TRANSACTION WITH HEURISTIC OUTCOME DESCRIBED ON THIS SERVER BY URID: ' *string* ' XID FORMATID: ' *string* ' XID GTRID: ' *string* ' XID BQUAL: ' *string* ' BECAUSE THE SUPERIOR COORDINATOR FOR SERVER *string* ON HOST *string*: *dstring* HAS NOT INVOKED FORGET ON THE REGISTERED RESOURCE.

Tips for RRS Operations

See *z/OS MVS Programming: Resource Recovery*, for RRS operations guidelines.

Tips for RRS operations:

- If you have configured your log streams to the coupling facility, then monitor your log streams to ensure offload is not occurring. RRS will perform better if its recovery logs do not offload.

Note: Proper sizing of the RRS logs is important. Too small and you get reduced throughput since logger is off-loading the logs too frequently. Too large and you could overflow your coupling facility.

- Keep the main and delayed (only contains active or live data) logs in your coupling facility. Make sure the CF definitions don't overflow.

Note: A commit cannot occur until the log record is written.

- Until you stabilize your workloads, it is a good idea to use the archive log. If you have an archive log configured, RRS will unconditionally use it. However, there is a performance penalty for using it.

Resolving InDoubt units if you receive a BBOT00xxW message

If you receive a BBOT00xxW message, you must use RRS panels to view the outcome of other branches in the transaction and set the outcomes of InDoubt units to match the outcome of those other branches.

Before you begin

When a BBOT00xxW message is displayed, there is a possibility of an heuristic outcome. See *z/OS MVS Programming: Resource Recovery* for more information on how to use the RRS panels and what administrative access (RACF access to the facility class, for example) is needed to resolve URs and remove interests.

About this task

Perform the following steps to remove an expression of interest in this UR if you receive any of the following error messages:

- Messages BBOT0019W and BBOT0020W, which appear together, indicate that this server could not determine the outcome from its superior. Message BBOT0020W, describes the resource to which the product could not provide an outcome.
 - Message BBOT0021W, which indicates that a server has determined the transaction outcome but has not been able to communicate it to its subordinates.
 - Message BBOT0022W, which indicates that the transaction outcome was determined and communicated to the subordinate, but the subordinate resource has not been "forgotten."
1. Select option 3, "Display/Update RRS Unit of Recovery information" on the main RRS panel.
 2. To view the details of this URID, enter it in the "URID Pattern" field on the query panel. Press the **Enter** key to execute the query. The query results should display the UR. Take note of whether the state of this UR is InCommit, InBackout or InForget. In the column labeled S, enter v to display the details for this UR.
 3. Remove the OTS interest. The RRS Unit of Recovery Details panel opens. Near the bottom of the panel, find the Expressions of Interest heading. This heading is followed by one or more rows that represent each individual expression of interest in this UR. Complete these steps to remove the OTS interest:
 - a. Find the row that represents the OTS interest. This row will have an RM name in the form of BSS00.xxx.yyy.IBM, where xxx is the system to which the server was configured and yyy is the specific server name.
 - b. Type r in the column labeled S to indicate that you want to remove this interest.
 - c. Press the **Enter** key to execute the query.
 4. Press the **Enter** key to confirm the removal of this interest. The RRS Remove Interest Confirmation panel opens. The RM name and UR identifier fields are pre-filled. Press the **Enter** key to confirm the removal of this interest.

Results

You know you are done when RRS marks the subordinate server as restart anywhere. Determine this by choosing option 1 under Browse and RRS log stream, and then choosing sub-option 4 under RRS Resource Manager Data log.

What to do next

Any subordinate nodes that restart and ask this server about this UR can not obtain this information. If you restart the server containing these nodes, they might be assigned an outcome that is different from the outcome of the transaction. You must manually resolve these nodes before you bring up the servers and start the server for which you just released the UR.

Resource Recovery Services Operations

This topic provides tips for using the z/OS Resource Recovery Services with this product.

Tips for RRS Operations

See *z/OS MVS Programming: Resource Recovery*, for RRS operations guidelines.

Tips for RRS operations:

- If you have configured your logstreams to the coupling facility, then monitor your log streams to ensure offload is not occurring. RRS will perform better if its recovery logs do not offload.

Note: Proper sizing of the RRS logs is important. Too small and you get reduced throughput since logger is off-loading the logs too frequently. Too large and you could overflow your coupling facility.

- Keep the main and delayed (only contains active or live data) logs in your coupling facility. Make sure the CF definitions don't overflow.

Note: A commit cannot occur until the log record is written.

- Until you stabilize your workloads, it is a good idea to use the archive log. If you have an archive log configured, RRS will unconditionally use it. However, there is a performance penalty for using it.

Recovering with JTA XAResource managers

When a JTA XAResource manager is enlisted in a global transaction, it cannot express an interest in the z/OS Resource Recovery Services unit of recovery (UR) like an RRS resource manager can. Instead, the product transaction service will save information in its RRS interest indicating that a JTA Resource Manager was enlisted in the transaction.

Purpose

When you look at the UR through the RRS panels, you will not see an interest for each XA transaction branch, as you would for a resource manager like DB2 or CICS interest.

Because of the differences between RRS and JTA XAResource Managers, there is a different set of errors that can occur when dealing with a JTA XAResource. The following sections describe errors you might see when recovering with a JTA XAResource Manager. Some of these errors are expected, while others may indicate that there is another type of problem, such as connectivity, that needs to be addressed.

This topic describes Peer Restart and Recovery messages that are unique to the z/OS environment.

Messages

- **BBOT0025D:** OTS HAS ENCOUNTERED A LOG DATA MISMATCH. REPLY CONTINUE IF THIS IS EXPECTED OR TERMINATE IF UNEXPECTED.

This message is issued when the restart epoch in the XA partner log for the product does not match the restart epoch in RRS. These logs must remain in sync to guarantee the atomic outcomes of distributed transactions.

If one or the other, but not both logs, were restored from a backup, a mismatch will occur. Since the XA partner log is maintained in the JVM, this error can also occur if the controller is started but then is canceled before the JVM is initialized. The RRS log stream will have been replayed before the XA partner log was initialized.

This message gives the operator an opportunity to cancel recovery and determine why the logs are not in sync. If the machine is not in production and data integrity is not an issue, the operator may reply **CONTINUE** and recovery will attempt to complete with the mismatched logs. However, the results of this response are unpredictable. If the operator replies **TERMINATE**, the application server will shut down, and the problem can be investigated before completing recovery.

- **BBOT0026I:** TRANSACTION SERVICE RESTART AND RECOVERY FOR SERVER %s IS STILL PROCESSING AN UNKNOWN NUMBER OF XA TRANSACTIONS.

This message is issued when the application server is unable to initiate contact with each JTA XAResource in its log. Since each JTA XAResource maintains its own logs, it is impossible to know how many transactions there are to recover. Look in the servant region for messages **WTRN0019**, and **WTRN0025**. These messages will help you determine what may be preventing the application server from communicating with these JTA XAResource Manager.

Creating application servers

During the installation process, the product creates a default application server, named server1. Most installations require several application servers to handle the application serving needs of their production environment. You can use the command-line tool or the administrative console to create additional application servers.

Before you begin

Determine if you want to use the application server that you are creating as part of a cluster. If this application server is going to be part of a cluster, you must use the Create a new cluster wizard instead of the Create a new application server wizard to create this application server. The topic *Adding members to a cluster* describes how to use the Create a new cluster wizard.

About this task

To create a new application server that is not part of a cluster, you can either use the Profile Management tool, the createApplicationServer, createWebServer, or createGenericServer wsadmin command, or you can use the administrative console.

If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous product level. This means that, for a period of time, you might be managing servers that are running at two different release levels in the same cell. However, when you create a new server definition, you must use a server configuration template, and that template must be created from a server instance that matches the version of the node for which you are creating the server.

There are no restrictions on what you can do with the servers running on the more current release level.

Complete the following steps if you want to use the administrative console to create a new application server that is not part of a cluster.

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > New**.

The Create a new application server wizard starts.

2. Select a node for the application server.

3. Enter a name for the application server. The name must be unique within the node.
4. Click **Next**.
5. Select a server template for the new server.

You can use a default application server template for your new server, or you can use the template that is optimized for development uses. The new application server inherits all of the configuration settings of the template server.

6. Click **Next**.
By default, this option is enabled. If you select this option, then you might need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. If you deselect this option, then ensure that the default port values do not conflict with other servers on the same physical machine.
7. Select **Generate unique HTTP ports** if you want the wizard to generate unique ports for the application server.
8. Optional: Click **Next** and specify a short name for the server.

The short name is also used as the JOBNAME for the server. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. For example, if default short names are already assigned to two other servers in the cell, the short name BBOS003 will be assigned to this server if you do not specify a short name when you create this server

Note: Make sure that you set up a RACF SERVER class profile that includes this short name.

9. Optional: Specify a generic short name for the server.
The generic short name for the server becomes the cluster transition name. If you do not specify a value for the generic short name field, the generic short name defaults to BBOCnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name. For example, if default generic short names are already assigned to three other servers in the cell, the generic short name BBOC004 is assigned to this server if you do not specify a generic short name when you create this server.

Note: Make sure that you set up a RACF SERVER class profile that includes this generic short name.

10. Click **Next**. Review the settings for the new server.
11. If you want to change any of the settings, click **Previous** until you return to a page where you can change that setting.
12. Click **Finish** when you do not want to make any additional changes.
13. Click **Review**, select **Synchronize changes with nodes**, and then click **Save** to save your changes.
14. Optional: Run the updateZOSStartArgs script to enable an application server to use the z/OS reusable ASID function, if it is not already enabled for the node that is associated with this application server.

This function enables an application server to reuse all ASIDs, including those that are associated with cross-process services.

Note: Before running this script, verify that you are running on z/OS Version 1.9 or higher, and that the reuse ASID function is enabled during the z/OS startup process. If the function is not enabled on z/OS, running this script has no affect on how ASIDs are handled.

Results

The new application server is in the list of servers on the administrative console Application servers page.

What to do next

This newly created application server is configured with default settings that are not displayed when you run the Create New Application Server wizard.

You can:

- In the administrative console, click **Servers > Server Types > WebSphere application servers** , and then click the name of this application server to view all of the configuration settings for this application server. You can then use this page to change some of the configuration settings for this server.

For example, if you do not need to have all of the sever components start during the server startup process, you might want to select **Start components as needed**, which is not automatically selected when a new server is created. When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of this product, make sure that those other products support this functionality before you select this property.

Note: The default addressing mode for a new server is 64-bit. You can deselect the Run in 64 bit mode field if you need to use 31-bit addressing mode. However, support for running a server in 31-bit mode is deprecated.

- Use server custom properties to modify the timer settings if you need to change the default timer settings for certain operations.
- Set the client.encoding.override Java virtual machine (JVM) argument to UTF-8 if you need to use multiple language encoding support in the administrative console.

Creating server templates

A server template is used to define the configuration settings for a new application server. When you create a new application server, you either select the default server template or a template you previously created, that is based on another, already existing application server. The default template is used if you do not specify a different template when you create the server.

About this task

To create a server template.

Use the **createServerTemplate** command for the AdminTask object.

If you do not create any additional server templates, the defaultZOS template is used as the template for the first cluster member. This template uses the default port assignments for the z/OS platform. If some of these ports are already defined for use elsewhere in your z/OS system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.

Results

Your new template is on the list of server templates that you can use to create a new application server or cluster member.

What to do next

You can perform one of the following actions to display a list of all of the server templates that are available on your system:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers >** , and then click **Templates**.
2. Issue the **listServerTemplates** wsadmin command.

Deleting server templates

The steps below describe how to delete a server template that you no longer need.

Use the **deleteServerTemplate** wsadmin command.

Results

The template you chose is removed from the list and cannot be used to create a new application server.

What to do next

You can perform one of the following actions to display a list of the server templates that are still available on your system:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and then click **Templates**.
2. Issue the **listServerTemplates** wsadmin command.

Managing application servers

You can use either the administrative console or command-line tools to manage your application servers.

Before you begin

If you plan to change the system clock, stop all the application servers, the node agent servers, the deployment manager server, the administrative agent server, the job manager server, and the location service daemon first. After you stop the servers and location service daemon, change the system clock, and then restart the servers and location service daemon. If you change the system clock on one system, you must ensure the clocks on all systems that communicate with each other and have WebSphere Application Server installed are synchronized. Otherwise, you might experience errors, such as security tokens no longer being valid.

About this task

Note: If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous release level. This means that, for a period of time, you might be managing servers that are running at different release levels in the same cell. In this mixed environment, some restrictions exist for what you can do with servers that are running at a previous release level. No restrictions exist for what you can do with the servers that are running on the newest release level.

You can perform the following steps to view and manage an application server from the administrative console.

1. In the administrative console click **Servers > Server Types > WebSphere application servers**.
The Application servers page lists the application servers in your environment and the status of each of these servers. You can use this page to complete the following actions:
 - Create additional servers.
 - Monitor running servers.
 - Control the status of a server.
 - Create a server template

- Delete a server. When you select a server for deletion, you must click **Delete** and **OK** before the server is deleted.

Note: If the server you are deleting has applications or modules mapped to it and is not part of a cluster, remap the modules to another server, or create a new server and remap the modules to the new server, before you delete this server. After a server to which modules are mapped is deleted, you cannot remap these modules to another server. Therefore, if you do not remap the modules to another server before you delete this server, you must uninstall all of the modules that were mapped to this server, and then reinstall them on a different server.

If the server you are deleting is part of a cluster, any application that is installed on this server is automatically installed on all of the other servers in the cluster. Therefore, deleting one cluster member does not affect the other cluster members, and the application remains installed in the cluster. Similarly when a new member is added to an existing cluster, any applications that are installed on the servers in that cluster are automatically installed on the new cluster member.

2. Click the name of a listed server to view or change the configuration settings for that server.

You can use this administrative console page to:

- Change the configuration settings for the selected server.

For example, if you do not need to have all of the sever components start during the server startup process, you might want to select **Start components as needed**, which is not automatically selected when a new server is created. When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

- View the status of applications running on the selected server. To view the status of applications running on this server, under Applications, click **Installed Applications**.

3. Click **Custom properties** to add new custom properties, update existing custom properties, or modify the timer settings if the current settings are causing timeout problems.
4. Click **Review**, select **Synchronize changes with Nodes**.
5. Click **Save** to save any configuration changes that you made.
6. If you made any configuration or custom property changes, start the application server, or stop and restart the application server if it is already running.

Results

When you click **Servers > Server Types > WebSphere application servers**, you can view the state of each server.

When you click **Servers > Server Types > WebSphere application servers > server_name**, you can view any configuration changes you made.

What to do next

You can deploy applications or components to your application servers.

Server collection

Use this topic to learn how to navigate within the administrative console to the pages where you can view information about the application servers, generic servers, Java message service (JMS) servers, and Web servers that are defined for your system.

You can use these respective administrative console pages to view the status of the listed servers. The status indicates whether a server is running, stopped, or encountering problems. You can also use these pages to perform the following actions for the listed servers:

- Select one or more of the listed servers, and then click **Start** to start those servers.
- Select one or more of the listed servers, and then click one of the following options to stop those servers:

STOP When you click this option, the normal server quiesce process is followed. This process allows in-flight requests to complete before the entire server process shuts down.

Immediate Stop

This option is only available for application servers.

When you click this button, the selected sever stops but the normal server quiesce process is not followed. This shutdown mode is faster than the normal server stop processing, but some application clients might receive exceptions if an in-flight request does not complete before the server process shuts down.

Terminate

This option is not available for Version 5 JMS servers.

You should only click **Terminate** if the server does not respond when you click **Stop**, or, **Immediate Stop** or when you issue the Stop or ImmediateStop commands. Some application clients can receive exceptions. Therefore, you should always attempt an immediate stop before clicking **Terminate**.

- Click **New** to create a new server.
- Click **Templates** to create a new server template.
- Select one or more of the listed servers, and then click **Delete** those servers. This option is not available for Version 5 JMS servers.

To view the Application servers page, in the administrative console page, click **Servers > Server Types > WebSphere application servers**. This page lists all of the application servers in the cell.

To view the Generic servers page, in the administrative console, click **Servers > Server Types > Generic servers**. This page lists all of the generic servers in the cell.

To view the Web servers page, in the administrative console, click **Servers > Server Types > Web servers**. This page lists all of the Web servers in your administrative domain. In addition to the previously mentioned actions, you can use this page to generate and propagate a Web server plug-in configuration file.

To view the Version 5 Java message service (JMS) servers page, in the administrative console page, click **Servers > Server Types > Version 5 JMS servers**. This page lists all of the JMS servers in the cell. Each JMS server provides the functions of the JMS provider for a node in your administrative domain. There can be, at most, one JMS server on each node in the administration domain, and any application server within the domain can access JMS resources served by any JMS server on any node in the domain.

Note: JMS servers apply only to WebSphere Application Server Version 5.x nodes. You cannot create a JMS server on a node that is not running WebSphere Application Server Version 5.x, but existing Version 5.x JMS servers continue to be displayed, and you can modify their properties. However, you cannot use this page to delete a Version 5.x JMS server.

Name

Specifies a logical name for the server. For WebSphere Application Server, server names must be unique within a node.

On the z/OS platform, this is sometimes called the long name. Server names must be unique within a node. If you have multiple nodes in a cluster, the server names must also be unique within the cluster. You cannot use the same server name within two nodes that are part of the same cluster. WebSphere uses the server name for administrative actions, such as referencing the server in scripting.

Node

Specifies the node on which the server resides.

Host Name

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the server.

Version

Specifies the version of the product on which the server runs.

Cluster Name





Specifies the name of the cluster to which the application server belongs. This field only displays when the **Include cluster members in the collection** console page preference is selected on the Applications server page.

If you have created clusters, and if the **Include cluster members in the collection** console page preference is selected, application servers that are cluster members are included in the list of application servers that displays on the Application servers page. These cluster members can be managed in the same manner as any of the other application servers in the list.

If the **Include cluster members in the collection** console page preference is not selected, application servers that are cluster members are not listed in the list of application servers that can be managed from this page.

Status

Specifies whether the server is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

	Started	The server is running.
	Partially stopped	The server is in the process of changing from a started state to a stopped state.
	Stopped	The server is not running.
	Unknown	The server status cannot be determined.

Application server settings

Use this page to configure an application server or a cluster member template. An application server is a server that provides services required to run enterprise applications. A cluster member template is the set of application server configuration settings that are assigned to new members of a cluster.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name**.

On the **Configuration** tab, you can change field settings. You can also click **Installed applications** to view the status of applications that are running on this server. On the **Runtime** tab, you can view read only information. The **Runtime** tab is available only when the server is running.

Name

Specifies a logical name for the server. Server names must be unique within a node. However, for multiple nodes within a cluster, you might have different servers with the same server name as long as the server and node pair are unique. You cannot change the value that appears in this field.

For example, a server named *server1* in a node named *node1* in the same cluster with a server named *server1* in a node named *node2* is allowed. However, you cannot have two servers named *server1* in the same node. The product uses the server name for administrative actions, such as referencing the server in scripting.

On the z/OS platform, this name is sometimes referred to as the long name.

Default server1

Short name

Specifies the short name of the server and must be unique within a cell. This field only displays for the z/OS platform. The short name is also the default z/OS job name and identifies the server to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), and started task control.

This field is optional, and only displays if you are running on z/OS. If you do not specify a value for the short name field, the short name defaults to BBOSnnn, where nnn is the first free number in the cell that can be used to create a unique short name. For example, if default short names are already assigned to two other servers in the cell, the short name BBOS003 will be assigned to this server if you do not specify a short name when you create this server. After the application server is created, you can change this generated short name to a name that conforms to your naming conventions.

The default values for the servant and adjunct jobnames are this short name with either an S, for the servant, or an A, for the adjunct, appended. If you must use an 8-character server short name, the servant and adjunct jobnames become 9-character names. Therefore, you must update the start command arguments for the servant and the adjunct process definitions to use the new 8-character server short name. The topic "Converting a 7-character server short name to 8 characters" describes how to perform this update.

If you specify a short name, the name:

- Must be one to eight characters in length. By default, z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number.
- Must be unique in the cell.

Generic short name

Specifies the generic short name of the server and must be unique within a cell. This field is optional, and only displays for the z/OS platform. The generic short name for the server becomes either the cluster transition name, if you are creating an unclustered server, or the cluster short name, if you are creating a clustered server.

If you do not specify a value for the generic short name field, the generic short name defaults to BBOCnnn, where nnn is the first free number in the cell that can be used to create a unique generic short name. For example, if default generic short names are already assigned to three other servers in the cell, the generic short name BBOC004 is assigned to this server if you do not specify a generic short name when you create this server.

If you specify a generic short name, the name:

- Must be one to eight characters in length.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell.

Run in development mode

Enabling this option might reduce application server start-up time because it changes some of the JVM settings, such as disabling bytecode verification, and reducing just-in-time (JIT) compiler compilation costs. Do not enable this setting on production servers. This setting is only available on an application server that is running in a Version 6.0 or later cell.

Specifies that you want to use the **-Xverify** and **-Xquickstart** JVM properties as startup values. Before selecting this option, add the **-Xverify** and **-Xquickstart** properties as generic arguments to the JVM configuration.

If you select this option, then you must save the configuration, and restart the server before this configuration change takes effect.

The default setting for this option is `false`, which indicates that the server does not start in development mode. Setting this option to `true` specifies that the server starts in development mode with settings that decrease server start-up time.

Data type	Boolean
Default	false

Parallel start

Select this field to start the server on multiple threads. This might shorten the startup time.

Specifies that you want the server components, services, and applications to start in parallel rather than sequentially.

The default setting for this option is `true`, which indicates that when the server starts, the server components, services, and applications start on multiple threads. Setting this option to `false` specifies that when the server starts, the server components, services, and applications start on a single thread, which might lengthen start-up time.

The order in which the applications start depends on the weights that you assign to them. Applications that have the same weight start in parallel.

To set the weight of an application, in the administrative console, click **Applications > Application Types > WebSphere enterprise applications > *application_name* > Startup behavior**, and then specify an appropriate value in the **Startup order** field. The more important an application is, the lower the startup order value should be. For example, you might specify a startup order value of 1 for your most important application, and a value of 2 for the next most important application. You might then specify a startup order of 3 for the next four applications because you want all four of those applications to start in parallel.

Data type	Integer
Default	1
Range	0 - 2147483647

Start components as needed

Select this property if you want the server components started as they are needed by an application that is running on this server.

When this property is selected, server components are dynamically started as they are needed. When this property is not selected, all of the server components are started during the server startup process. Therefore, selecting this option can improve startup time, and reduce the memory footprint of the server, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the server, are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

Note: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

Run in 64 bit JVM mode

Specifies that the application server runs in 64-bit mode, which is the default setting. Running in 64-bit mode provides additional virtual storage for user applications. This field only displays for the z/OS platform.

By default, the WebSphere Customization Tools are set up to start all of your application servers in 64-bit mode. However, you can change the settings for the WebSphere Customization Tools, such that all of your application servers start in 31-bit mode. You can also unselect this setting for a subset of your application servers if you only want specific application servers to start in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. Therefore, you can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, you should eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Access to internal server classes

Specifies whether the applications that are running on this server can access multiple server implementation classes.

If you select `Allow`, then applications can access many of the server implementation classes. If you select `Restrict`, then applications cannot access multiple server implementation classes. The applications get a `ClassNotFoundException` error if they attempt to access those classes.

Usually you should select `Restrict` for this property, because most applications use the supported APIs and do not need to access any of the internal classes. However, if your application requires the use of one or more of the internal server classes, select `Allow` as the value for this property.

The default value for this property is `Allow`.

Class loader policy

Select whether there is a single class loader to load all applications or a different class loader for each application.

Class loading mode

Specifies whether the class loader searches in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and the product class loaders is `Parent first`.

This field only applies if you set the `Class loader policy` field to `Single`.

If you select `Application` first, your application can override classes contained in the parent class loader, but this action can potentially result in `ClassCastException` or linkage errors if you have mixed use of overridden classes and non-overridden classes.

Process ID

The process ID for this server on the native operating system.

This property is read only. The system automatically generates the value.

Cell name

The name of the cell in which this server is running.

This property is read only.

Node name

The name of the node in which this server is running.

This property is read only.

State

The runtime start state for this server.

This property is read only.

Product information

This link under `Additional properties`, displays the product information for your installation of the product. This information includes the product name, ID, version, build date, and build level.

From the `Product Information` page, you can click on the following links for additional product information:

- `Components`, for a list of all of the components that are installed.
- `e-Fixes`, for a list of all of the service updates that are installed.
- `Extensions`, for a list of the extensions that are installed.
- `History report`, for a detailed report of all installation events that have occurred since the product was installed, such as the installation of a specific service level.
- `Product report`, for a detailed report of the versions of the product that are installed.
- `PTFs`, for a list of all of PTFs that are installed.

Ports collection

Use this page to view and manage communication ports used by run-time components running within a process. Communication ports provide host and port specifications for a server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name* > Communications > Ports**.

This page displays only when you are working with ports for application servers.

Port Name:

Specifies the name of a port. Each name must be unique within the server.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, or administrative service).

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Transport Details:

Provides a link to the transport chains associated with this port. If no transport chains are associated with this port, the string "No associated transports" appears in this column.

Ports settings:

Use this to view and change the configuration for a communication port used by run-time components running within a process. A communication port provides host and port specifications for a server.

If you are running on z/OS, you can view this administrative console page by clicking one of the following paths:

- **Servers > Server Types > WebSphere application servers > *server_name* > Ports > *end_point_name***
- **Servers > Server Types > JMS servers > *server_name* > Ports > *end_point_name***

Port Name:

Specifies the name of the port. The name must be unique within the server.

Note that this field displays only when you are defining a port for an application server. You can select either:

Well-known Port

When you select this option , you can select a previously defined port from the drop down list

User-defined Port

When you select this option, you must create a port with a new name by entering the name of the new port in the text box

Data type	String
------------------	--------

The following ports apply to the z/OS platform only.

End point

JMSSERVER Queued Address

Description

Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Queued Address port is the listener port used for full-function JMS-compliant, publish/subscribe support. The default Queued Address port number is 5558.

Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.

End point**JMSSERVER Direct Address****Description**

Specifies the host and port number used to configure the WebSphere JMS Provider topic connection factory. The JMS Server Direct Address port is the listener port used for direct TCP/IP connection (non-transactional, nonpersistent, and nondurable subscriptions only) for publish/subscribe support. The default Direct Address port number is 5559.

Since the queue manager and queue broker for the WebSphere JMS Provider are configured outside the administrative console, changes to this port require corresponding configuration changes to the queue manager and queue broker.

Host:

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a resource (such as the naming service, administrative service, or JMS broker).

For example, if the host name is `myhost`, the fully qualified DNS name can be `myhost.myco.com` and the IP address can be `155.123.88.201`.

Host names on the ports can be resolvable names or IP addresses. The server will bind to the specific host name or IP address that is supplied. That port will only be accessible through the IP address that is resolved from the given host name or IP address. The IP address may be of the IPv4 (Internet Protocol Version 4) format for all platforms, and IPv6 (Internet Protocol Version 6) format on specific operating systems where the server supports IPv6.

Note: If your TCP/IP network is set up to use distributed dynamic virtual IP addresses (DVIPAs), and if the node agent is in the process of starting the application server, TCP/IP waits until the JVM TCP/IP timeout period expires before notifying the node agent that the target application server is not responsive.

Data type

String

Default

* (asterisk)

Port:

Specifies the port for which the service is configured to accept client requests. The port value is used in conjunction with the host name.

Port numbers in the server can be reused among multiple ports as long as they have host names that resolve to unique IP addresses and there is not a port with the same port number and a wildcard (*) host name. A port number is valid in the range of 0 and 65535. 0 specifies that the server should bind to any ephemeral port available. Specifying the wildcard value is equivalent to specifying the loopback address or `127.0.0.1`.

Note: Port sharing cannot be created using the administrative console. If you need to share a port, you must use `wsadmin` commands to define that port. You must also make sure that the same discrimination weights are defined for all of the transport channels associated with that port.

Protocol channels only accept their own protocol. However, application channels usually accept anything that reaches them. Therefore, for application channels, such as `WebContainer`, or `Proxy`, you should specify larger discrimination weights when sharing levels with protocol channels, such

as HTTP or SSL. The one exception to this rule is if you have application channels that perform discrimination tests faster than the protocol channels. For example, a JFAP channel is faster at deciding on a request than the SSL protocol channel, and should go first for performance reasons. However, the WebContainer and Proxy channels must always be last because they accept everything that is handed to them.

Data type Integer
Default None

Note: The following table lists server endpoints and their respective port ranges. In contrast to an i5/OS or distributed platform environment, for a z/OS environment, the ORB_LISTENER_ADDRESS and the BOOTSTRAP_ADDRESS must specify the same port.

Endpoint (port)	Acceptable values for the port field
BOOTSTRAP_ADDRESS	0 - 65535
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	Not supported on z/OS
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	Not supported on z/OS
DATAPOWERMGR_INBOUND_SECURE	1 - 65536
DCS_UNICAST_ADDRESS	1 - 65536
DRS_CLIENT_ADDRESS	1 - 65536
ORB_LISTENER_ADDRESS	1 - 65535 (If 0 is specified, the server starts on any available port and does not use the location service daemon)
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	Not supported on z/OS
SIB_ENDPOINT_ADDRESS	1 - 65536
SIB_ENDPOINT_SECURE_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_ADDRESS	1 - 65536
SIB_MQ_ENDPOINT_SECURE_ADDRESS	1 - 65536
SOAP_CONNECTOR_ADDRESS	1 - 65536
WC_adminhost	1 - 65536
WC_adminhost_secure	1 - 65536
WC_defaulthost	1 - 65536
WC_defaulthost_secure	1 - 65536
ORB_SSL_LISTENER_ADDRESS	0 - 65535 (0 specifies that the server should bind to any ephemeral port that is available.)

Custom property collection

Use this page to view and manage arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties.

The administrative console contains several Custom Properties pages that work similarly. To view one of these administrative pages, click one of the **Custom properties** links.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in the application server.

Value:

Specifies the value paired with the specified name.

Description:

Provides information about the name-value pair.

Custom property settings:

Use this page to configure arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties. Defining a new property enables you to configure a setting beyond that which is available in the administrative console.

Note: Setting custom properties at the server level is deprecated. However, you can specify a custom property for a server or the deployment manager as a WebSphere variable. Server scoped WebSphere variables still override any settings specified at the node scope, or higher, and are added to the `was.env` file.

To set a custom property for either the deployment manager, or an application server, as an environment variable, in the administrative console, click **Environment > WebSphere variables**, select the deployment manager or server from the pull-down list of available servers, nodes and cells, and then click **New**.

To view this administrative console page, click one of the following paths:

- For an application server, click **Servers > Server Types > WebSphere application servers > server_name**. Then, in the Server Infrastructure section, click **Administration > Custom properties**.
- For a JMS server, click **Servers > Server Types > JMS servers > server_name**. Then, in the Server Infrastructure section, click **Administration > Custom properties**.
- For a deployment manager, click **System Administration > Deployment manager**, and then in the Server Infrastructure section, click **Java and process management > Process definition > Java virtual machine > Custom properties**.

You can then click **New** to, create a new custom property, click on the name of an existing property to change its settings, or click **Delete** to delete an existing property.

Name:

Specifies the name (or key) for the property.

Each property name must be unique. If the same name is used for multiple properties, the value specified for the first property is used.

Do not start your property names with `was.` because this prefix is reserved for properties that are predefined in the product.

Data type String

Value:

Specifies the value paired with the specified name.

Data type String

Description:

Provides information about the name and value pair.

Data type String

Native processes

Use this page to view and modify properties of the JMS Integral Provider native processes.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name**. Then, in the Server Infrastructure section, click **Administration > Server components > JMS servers**.

Short name:

Specifies the short name of the JMS queue manager.

The name is 1-4 characters, alphanumeric or national language. It cannot start with a numeric.

The system assigns a unique default short name for the JMS queue manager.

Data type String

Command Prefix:

Specifies the subsystem command prefix for the JMS queue manager.

This field is read only because it is only configured using either the Profile Management Tool, or the zpmt command.

Data type String

WebSphere MQ CRA settings

Use this panel to configure the Control Region Adjunct (CRA) settings for an application server or a cluster member template.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name > Messaging > WebSphere MQ CRA settings**.

On the **Configuration** tab, under General Properties, you can change the CRA property setting.

Start CRA:

Enables the CRA process to start.

Data type	Checkbox
Default	Cleared
Range	Cleared Do not start the CRA process.
	Selected Start the CRA process.

Server component collection

Use this page to view information about and manage the types of server components that a specific application server uses during application processing. The list of server components varies according to the type of applications a specific application server processes.

For example, SIP Container might be listed as a server component for an application server that handles Session Initiation Protocol (SIP) requests, while EJB Container might be listed as a server component for an application server that handles Enterprise JavaBeans (EJB) requests. However, Messaging Server might be listed as a server component for both application servers.

You can also use this page to manage the settings for these server component, as they relate to request processing. In particular, you can specify either started or stopped as the initial state for the server component when the server process starts.

To view this administrative console page, click **System administration > Deployment Manager***server_name*. Then, in the Server Infrastructure section, click **Administration > Server components**.

To view this administrative console page for a node agent, click **System administration > Node agents > node_agent_name**. Then, in the Server Infrastructure section, click **Administration > Server components**.

Type:

Specifies the server component type, such as Name Server or Messaging Server.

Server component settings:

Use this page to view or configure a server component instance.

To view this administrative console, click **Servers > Server Types > WebSphere application servers > server_name**. Then, in the Server Infrastructure section, click **Administration > Server components > server_component_name**.

Name:

Specifies the name of the component.

Data type	String
------------------	--------

Initial State:

Specifies the desired state of the component when the server process starts. The options are: *Started* and *Stopped*. The default is *Started*.

Data type	String
Default	Started

Server instance settings

Use this page to view and manage servant instance settings. These settings control the number of servant processes that are allowed.

To view this page, in the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**. Next, in the Server Infrastructure section, click **Java and process management > Server instance**

Multiple Instances Enabled:

Specifies whether multiple servant process instances are enabled for this server.

When the Multiple Instances Enabled setting is disabled, the server has exactly one servant process instance. When the Multiple Instances Enabled setting is enabled:

- The Minimum number of instances setting determines the minimum number of process instances that are active for this server.
- The Maximum number of instances setting determines the maximum number of process instances that can be active for this server.

The z/OS Workload Manager dynamically determines the actual number of servant process instances.

Minimum number of instances:

Specifies the minimum number of servant process instances to activate.

Data type	Integer
Range	1 to 20, inclusive

Maximum number of instances:

Specifies the maximum number of servant process instances to activate.

Data type	Integer
Range	Any value. If zero is specified, the number of instances is unlimited.

Core group service settings

Use this page to set up the application server properties that relate to core groups.

To view this administrative console page, click **Servers > Server Types > Application servers > server**. Then in the Additional Properties section, select **Core group service**.

Click **Save** to save and synchronize your changes with all managed nodes.

Enable service at server startup

Select if you want the core group service, also known as the high availability manager service, to start on this process when the server starts. The core group service must be started before high availability functions, such as routing, and failover, work properly.

Note: The default value for this setting is selected Before disabling the core group service for a server process, make sure that none of the components that this process uses require high availability functions.

Default	Core group service starts when the server starts.
----------------	---

Core group name

Specifies the name of the core group that contains this application server as a member. To move a server to a different core group, in the administrative console, click **Servers > Core groups > Core group settings > core_group > Core group servers**.

Data type	String
------------------	--------

Allow activation

Select if high availability group members can be activated on this application server.

Is alive timer

Specifies the time interval, in seconds, at which the high availability manager will check the health of all of the active high availability group members that are running in this application server process. An active group member is a member that is able to accept work. If a group member fails, the application server on which the group member resides is restarted. If -1 is specified, the timer is disabled. If 0 (zero) is specified, the default value of 120 seconds is used.

Note: The value specified for this property can be overridden for the high availability groups using a particular policy if the `Is alive timer` property for that policy specifies a different time interval. If the `Is alive timer` setting specified for a policy is greater than 0 (zero), the high availability manager uses that time interval, instead of the one specified at this level, when determining how frequently it should check the health of a high availability group member using that particular policy.

Data type	Any integer between -1 and 600, inclusive
Default	120 seconds

Transport buffer size

Specifies the buffer size, in megabytes, of the underlying group communication transport. The minimum buffer size is 10 megabytes.

Data type	String
Default	10 megabytes

Running servers in 31-bit mode

When you create a new application server, it is automatically configured to run in 64-bit mode. You can deselect the **Run in 64 bit JVM mode** setting if you need to run the server in 31-bit mode. However, whenever possible, leave your servers configured to run in 64-bit mode because support for running servers in 31-bit mode is deprecated. If you have any servers, that you migrated from a previous version of the product, that are running in 31-bit mode, consider reconfiguring them to run in 64-bit mode.

Note: Because support for running a server in 31-bit mode is deprecated, whenever you start a server that is configured to run in 31-bit mode, you receive the following message in your system log, where *server_name* is the name of the server that is running in 31-bit mode:

```
BB000340W: 31-BIT MODE IS DEPRECATED FOR THE APPLICATION SERVER RUNNING ON THE Z/OS OPERATING SYSTEM.  
CONSIDER USING 64-BIT MODE FOR server_name AS AN ALTERNATIVE.
```

When a server runs in 31-bit mode, the following conditions exist:

- Each server address space can access a maximum of 2 gigabytes of virtual memory.
- The 31-bit JVM, that is located in the *app_server_root/java* directory, is used for the server.

When a server runs in 64-bit mode, the following conditions exist:

- Each server address space can access a maximum of 16 exabytes (16 thousand million gigabytes) of virtual memory.
- The 64-bit JVM, that is located in the *app_server_root/java64* directory, is used for the server.

System requirements

Before starting to use a server that is configured to run in 64-bit mode, make sure that:

- The AMODE parameter can be used to specify a particular addressing mode, either 31-bit or 64-bit, for the server. This parameter can also be set to a value of 00, which indicates that the server is to use the configured addressing mode. In the generated procedures, 00 is the default value for the AMODE parameter.

If you convert a 31-bit server to the 64-bit addressing mode, make sure that your automation does not specify AMODE=31 on the MVS START command. If a server is started with an AMODE value that does not match the configured addressing mode, the server does not start.

Note: If the AMODE parameter is omitted, or set to 00 on the MVS START command, then the server starts in the currently configured addressing mode. If any other AMODE parameter is specified, that parameter must match the currently configured addressing mode. If the parameter does not match the currently configured addressing mode, the server terminates with one of the following error messages:

```
BB000336E START OF WEBSPPHRE FOR Z/OS PROCESS FAILED BECAUSE INPUT
AMODE 31 DOES NOT MATCH CONFIGURED AMODE 64
```

```
BB000336E START OF WEBSPPHRE FOR Z/OS PROCESS FAILED BECAUSE INPUT
AMODE 64 DOES NOT MATCH CONFIGURED AMODE 31
```

- The REGION parameter on the JCL JOB or EXEC statement, and the MEMLIMIT parameter on the JCL JOB or EXEC statement, or in the SMFPRMxx parmlib member, determines the amount of virtual storage that a particular server can obtain. If you do not specify REGION=0M in the server cataloged procedure, you must use the MEMLIMIT parameter to set a virtual storage limit larger than the 2 gigabyte limit associated with 31-bit mode.

Note: Make sure that your system exits do not limit the address space size for 64-bit servers inappropriately.

- Running servers in 64-bit mode requires additional auxiliary storage, in the form of either expanded storage or page data set space. Before running servers in 64-bit mode, review your page data set allocations for each z/OS target system, and add additional page data set space as needed. Also monitor paging and page data set utilization to ensure that the allocated auxiliary storage is sufficient.

For more information, see the following z/OS publications:

- *MVS Initialization and Tuning Reference*, SA22-7592
- *MVS Programming: Extended Addressability Guide*, SA22-7614

Converting a migrated server to run in 64-bit mode

Before converting an application server from 31-bit to 64-bit mode, complete the following actions:

- Verify that your system meets the requirements specified in the Systems requirements section.
- Verify that all applications that you plan to run on this application server are updated to use 64-bit native code and DLLs.

The DLLs and other native code that your applications call must match the addressing mode of the server on which the applications are running. If you convert an existing application server from 31-bit mode to 64-bit mode, you must change any Java applications containing native code, for example, C++ or Cobol, that you plan to run on the converted server, to run in 64-bit mode. Java applications can use the com.ibm.vm.bitmode Java property to determine the mode in which the server is running, and then load the correct 31-bit or 64-bit DLL to the native code.

An abend might occur if a server that is running in 64-bit mode tries to invoke an application that contains a 31-bit native module. Similarly, An abend might occur if a server that is running in 31-bit mode tries to invoke an application that contains a 64-bit native module.

For more information about converting language-environment (LE) applications to run in 64-bit mode, see the z/OS publication *Language Environment Programming Guide for 64-bit Virtual Addressing Mode*, SA22-7569.

To convert an application server from 31-bit mode to 64-bit mode, in the administrative console select the **Run in 64 bit JVM mode** option on the configuration settings page for that application server, and change the minimum and maximum JVM heap sizes to values that are appropriate for a 64-bit process. Similarly to convert an application server from 64-bit mode to 31-bit mode, deselect the **Run in 64 bit JVM mode** option on the configuration settings page for that application server, and change the minimum and maximum JVM heap sizes to values that are appropriate for a 31-bit process.

If you use the MVS START command to start a 64-bit server, make sure that the AMODE parameter is set to 00 or 64, or is allowed to default to 00, on the START command. For example, you might issue one of the following commands:

```
S BB07ACR,JOBNAME=BBOS001,ENV=BB0BASE.BB0NODE.BBOS001,AMODE=64
```

```
S BB07ACR,JOBNAME=BBOS001,ENV=BB0BASE.BB0NODE.BBOS001
```

The startServer.sh command, and the administrative console, automatically add the AMODE=64 parameter when they are used to start a 64-bit application server.

Converting a migrated deployment manager to run in 64-bit mode

Before converting a deployment manager from 31-bit to 64-bit mode, complete the following actions:

- Verify that your system meets the requirements specified in the Systems requirements section.
- Verify that all applications that you plan to run on the deployment manager are updated to use 64-bit native code and DLLs.

The DLLs and other native code that your applications call must match the addressing mode of the server on which the applications are running. If you convert an existing application server from 31-bit mode to 64-bit mode, you must change any Java applications containing native code, for example, C++ or Cobol, that you plan to run on the converted server, to run in 64-bit mode. Java applications can use the com.ibm.vm.bitmode Java property to determine the mode in which the server is running, and then load the correct 31-bit or 64-bit DLL to the native code.

An abend might occur if a server that is running in 64-bit mode tries to invoke an application that contains a 31-bit native module. Similarly, An abend might occur if a server that is running in 31-bit mode tries to invoke an application that contains a 64-bit native module.

For more information about converting language-environment (LE) applications to run in 64-bit mode, see the z/OS publication *Language Environment Programming Guide for 64-bit Virtual Addressing Mode*, SA22-7569.

To convert a deployment manager from 31-bit mode to 64-bit mode, in the administrative console, select the **Run in 64 bit JVM mode** option on the configuration settings page for the deployment manager, and change the minimum and maximum JVM heap sizes to values that are appropriate for a 64-bit process.

If you use the MVS START command to start a 64-bit deployment manager, make sure that the AMODE parameter is set to 00 or 64, or is allowed to default to 00, on the START command. For example, you might issue one of the following commands:

```
S BB07DCR,JOBNAME=BBODMGR,ENV=PLEXA.PLEXABBOCELL.BBODMGR.BBODMGR,AMODE=64
```

```
S BB07DCR,JOBNAME=BBODMGR,ENV=BBOCELL.BBODMGR.BBODMGR
```

The startServer.sh command, and the administrative console, automatically add the AMODE=64 parameter when they are used to start a 64-bit deployment manager.

Changing the values of variables referenced in BBOM0001I messages

BBOM0001I messages are issued during server startup, and indicate the product configuration settings. Unless otherwise indicated, these variables apply to all application servers, including the deployment managers, node agents, and JMS servers.

Before you begin

Not all of these settings can be changed. However, the settings that you can change must be changed using the administrative console. Any changes that you make directly to the was.env file for a specific server are discarded the next time that you use the administrative console to make a configuration change.

Use the following table to determine which product variable, custom property or administrative console field must be updated to change the value of a specific internal variable.

Table 4. Mapping internal variables reference in BBOM0001I messages to external WebSphere variable, custom property or administrative console fields

Internal Variable Name	How to Change Indicated Value	Comments
adjunct_region_libpath (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Adjunct > Environment entries > LIBPATH , and specify a different value.	Specifies the libpath for the adjunct process' JVM.
cell_name	User cannot change.	Initially specified during installation and customization.
cell_short_name	User cannot change.	Initially specified during installation and customization.
client_protocol_resolve_name	User cannot change.	No longer used. Message will not appear for V5.1 and higher.
client_protocol_resolve_port	User cannot change.	No longer used. Message will not appear for V5.1 and higher.
client_ras_logstream_name	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the <code>client_ras_logstreamname</code> property and specify a different value.	See Application server z/OS custom properties for additional information.
com_ibm_security_SAF_EJBROLE_Audit_Messages_Suppress	In the administrative console, click Security > Security administration > Applications > Infrastructure . In the Additional Properties section, click Custom properties > New . Add the <code>com_ibm_security_SAF_EJBROLE_Audit_Messages_Suppress</code> property, and specify a different value.	
com_ibm_DAEMON_claim_ssl_sys_v3_timeout	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS location service .	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_claimClientAuthentication	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS location service .	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_claimKeyringName	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS location service .	Select an existing alias or create a new SSL Configuration Repertoire.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

com_ibm_DAEMON_claimSecurityCipherSuiteList	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS location service .	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_DAEMON_claimSecurityLevel	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS location service .	Select an existing alias or create a new SSL Configuration Repertoire.
com_ibm_HTTP_claim_ssl_sys_v2_timeout	User cannot change.	This variable has been deprecated.
com_ibm_HTTP_claim_ssl_sys_v3_timeout	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports > ,ssl_transport .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claim_sslEnabled	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports,ssl_transport .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimClientAuthentication	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimKeyringName	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimSecurityCipherSuiteList	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_HTTP_claimSecurityLevel	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports .	Transport option only appears if you have an HTTP transport defined for your system.
com_ibm_Server_Security_Enabled	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Security > Server security .	Select an existing alias or create a new SSL Configuration Repertoire. This setting overrides the setting specified using the Security > Global Security > enabled/disabled field in the administrative console.
control_region_classpath	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Java Virtual Machine , and specify the classpath to be appended.	Specifies the classpath used by the controller's JVM.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

control_region_http_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_http_queue_timeout_percent property and specify a different value.	Indicates the percentage of the HTTP dispatch time limit that should be used as the maximum amount of time that an HTTP request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_https_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_https_queue_timeout_percent property and specify a different value.	Indicates the percentage of the HTTPS dispatch time limit that should be used as the maximum amount of time that an HTTPS request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_iiop_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_iiop_queue_timeout_percent property and specify a different value.	Indicates the percentage of the IIOp dispatch time limit that should be used as the maximum amount of time that an IIOp request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_jvm_localrefs		Should only be used under the direction of IBM support.
control_region_jvm_logfile	User cannot change.	Specifies the file to which the controller's JVM will write messages.
control_region_jvm_properties_file	User cannot change.	Is dynamically created.
control_region_libpath	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition , and specify the libpath.	
control_region_mdb_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_mdb_queue_timeout_percent property and specify a different value.	Indicates the percentage of the MDB dispatch time limit that should be used as the maximum amount of time that an MDB request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_mdb_request_timeout (application server only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_mdb_request_timeout property and specify a different value.	See Application server z/OS custom properties for additional information.
control_region_security_enable_trusted_applications (application server only)	In the administrative console, click Security > Global security > Custom properties . Select the EnableTrustedApplications property, and specify a new value.	

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

control_region_sip_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_sip_queue_timeout_percent property and specify a different value.	Indicates the percentage of the SIP dispatch time limit that should be used as the maximum amount of time that a SIP request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_sips_queue_timeout_percent	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_sips_queue_timeout_percent property and specify a different value.	Indicates the percentage of the SIPS dispatch time limit that should be used as the maximum amount of time that a SIPS request can spend on the workload management (WLM) queue. See Application server z/OS custom properties for additional information.
control_region_ssl_thread_pool_size		Should only be changed under the direction of IBM Support personnel.
control_region_thread_pool_size		Should only be changed under the direction of IBM Support personnel.
control_region_thread_stack_size		Should only be changed under the direction of IBM Support personnel.
control_region_timeout_delay (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_timeout_delay property and specify a different value.	See Configuring server properties for additional information.
control_region_timeout_dump_action	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_timeout_dump_action property and specify JAVADUMP, javadump, SVCDUMP, or svcdump for the value.	Indicates whether a Java core dump or an SVC dump should be taken whenever a timeout occurs for work that has been dispatched to a servant. See Application server z/OS custom properties for additional information
control_region_timeout_dump_action_session	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_timeout_dump_action_session property and specify JAVADUMP, javadump, SVCDUMP, or svcdump for the value.	Indicates whether a Java core dump or an SVC dump should be taken whenever a timeout occurs for an HTTP, HTTPS, SIP, or SIPS request that has been dispatched to a servant. See Application server z/OS custom properties for additional information
control_region_timeout_save_last_servant	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the control_region_timeout_save_last_servant property and specify a different value.	Indicates whether the last available servant should be abended if a timeout situation occurs on that servant, or the last available servant should remain active until a new servant is initialized. See Application server z/OS custom properties for additional information

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

control_region_use_java_g	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Java Virtual Machine , and select Debug Mode .	Indicates if controller's JVM should use the debug JVM (java_g). Should only be changed under the direction of IBM Support personnel.
control_region_wlm_dispatch_timeout (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Container Settings, click Container Services > ORB Service and specify a new value in the WLM timeout field.	
daemon_group_name	User cannot change.	
daemon_protocol_iiop_listenIPAddress	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the daemon_protocol_iiop_listenIPAddress property, and specify either * to bind all, or the IP name to specify bind-specific support.	Allows you to restrict the IP addresses to which the location service daemon binds. You set this variable in your cell-level variables.xml file.
daemon_start_command	User cannot change.	
daemon_start_command_args	User cannot change.	
daemon_wlmable	User cannot change.	
daemonInstanceName	User cannot change.	
daemonName	User cannot change.	
default_internal_work_transaction_class	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the default_internal_work_transaction_class property, and specify a different value.	See Application server z/OS custom properties for additional information.
nls_language	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the nls_language property, and specify a different value.	See Application server z/OS custom properties for additional information.
node_name	User cannot change.	
node_short_name	User cannot change.	
nonauthenticated_clients_allowed	In the administrative console, click Security > Global security . Under Authentication, click Authentication Protocol > zSAS authentication .	
protocol_accept_http_work_after_min_srs	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_accept_http_work_after_min_srs property, and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_accept_iiop_work_after_min_srs	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_accept_iiop_work_after_min_srs property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_response_failure	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_bboc_log_response_failure property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_bboc_log_return_exception	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_bboc_log_return_exception property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_giop_level_highest	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_giop_level_highest property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_backlog	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_http_backlog property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_large_data_inbound_buffer	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_http_large_data_inbound_buffer property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_large_data_response_buffer	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_http_large_data_response_buffer property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_max_connect_backlog	User cannot update.	No longer used.
protocol_http_max_keep_alive_connections	User cannot update.	No longer used
protocol_http_timeout_output (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports > Custom properties . Select the ConnectionResponseTimeout property and specify a new value.	See HTTP transport custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_http_timeout_output_recovery (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_http_timeout_output_recovery property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_transactionClass	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_http_transactionClass property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_http_transport_class_mapping_file	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > z/OS additional Settings .	Note: This variable is deprecated. When possible, use a workload classification document file instead of this variable. See “Classifying z/OS workload” on page 303 for more information.
protocol_https_backlog	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_https_backlog property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_max_connect_backlog	User cannot change.	No longer used.
protocol_https_max_keep_alive_connections	User cannot change.	No longer used.
protocol_https_timeout_output (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web container settings > Web container > HTTP transports > ssl_transport	See HTTP transport custom properties for additional information.
protocol_https_timeout_output_recovery (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_https_timeout_output_recovery property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_https_transactionClass	User cannot change. The value specified for the protocol_http_transport_class_mapping_file variable is also used for this variable.	
protocol_iiop_backlog_ssl	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_iiop_backlog_ssl property, and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

protocol_iiop_backlog	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_iiop_backlog property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_iiop_daemon_listenIPAddress	In the administrative console, click System Administration > Node groups > Sysplex node group > z/OS Location Service . Specify the new IP address.	
protocol_iiop_daemon_port	In the administrative console, click System Administration > Node groups > sysplex node group > z/OS Location Service .	
protocol_iiop_daemon_port_ssl	In the administrative console, click System Administration > Node groups > sysplex node group > z/OS Location Service .	
protocol_iiop_no_local_copies	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Container Settings, click Container Services > ORB Service .	
protocol_iiop_propagate_unknown_service_ctxs	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_iiop_propagate_unknown_service_ctxs property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_sip_timeout_output_recovery (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_sip_timeout_output_recovery property, and specify a different value.	See Application server z/OS custom properties for additional information.
protocol_sips_timeout_output_recovery (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the protocol_sips_timeout_output_recovery property, and specify a different value.	See Application server z/OS custom properties for additional information.
ras_debugEnabled		Should only be changed under the direction of IBM Support personnel.
ras_default_msg_dd	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_default_msg_dd property, and specify a different value.	
ras_dumpoptions_dumptype		Should only be changed under the direction of IBM Support personnel.
ras_hardcopy_msg_dd	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_hardcopy_msg_dd property, and specify a different value.	

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

ras_log_logstreamName	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_log_logstreamName property, and specify a different value.	
ras_minorcode_action		Should only be changed under the direction of IBM Support personnel.
ras_stderr_ff_interval	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_stderr_ff_interval property, and specify a different value.	
ras_stdout_ff_interval	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_stdout_ff_interval property, and specify a different value.	
ras_time_local	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_time_local property, and specify a different value.	See Application server z/OS custom properties for additional information.
ras_trace_basic		Should only be changed under the direction of IBM Support personnel.
ras_trace_ctraceParms		Should only be changed under the direction of IBM Support personnel.
ras_trace_defaultTracingLevel		Should only be changed under the direction of IBM Support personnel.
ras_trace_detail		Should only be changed under the direction of IBM Support personnel.
ras_trace_exclude_specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_minorCodeTraceBacks		Should only be changed under the direction of IBM Support personnel.
ras_trace_outputLocation	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_trace_outputLocation property, and specify a different value.	
ras_trace_specific		Should only be changed under the direction of IBM Support personnel.
ras_trace_BufferCount	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_trace_BufferCount property, and specify a different value.	

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

ras_trace_BufferSize	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the ras_trace_BufferSize property, and specify a different value.	
read_license_agreement	User cannot change.	Indicates that the server's initialization code will verify license agreement. The default is 1. If this variable is not set to 1, the server will not initialize.
security_SMF_record_first_auth_user	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the pull-down list of available nodes and cells, and then click New . Add the security_SMF_record_first_auth_user property, and set it equal to 1 to fill the SM120CRE field with the ID of the first authenticated user. By default, the property is set to 0, and the SM120CRE field is filled with the ID under which the server activity began. This is often the guest ID.	If no authentication for a request is required, then the SM120CRE field will NOT contain an authenticated user ID for that request. Instead, it will contain an unauthenticated ID, typically the guest account ID or the WebSphere server ID.
security_sslKeyring	User cannot change.	No longer used.
security_zOS_domainName	User cannot change.	Set during customization.
security_zOS_domainType	User cannot change.	Set during customization.
security_zSAS_ssl_repertoire	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Server security > zSAS Transport > SSL Settings . Select a different repertoire.	
security_EnableRunAsIdentity	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the security_EnableRunAsIdentity property, and specify a different value.	See Application server z/OS custom properties for additional information.
security_sslType1	User cannot change.	No longer used.
server_configured_system_name	User cannot change.	Specifies the name of the system to which the server instance was originally configured.
server_generic_short_name	If the server is clustered, this value is the cluster's short name. If the server is not clustered, this value is the value specified on the server custom property, ClusterTransitionName . Either way, this value can be changed using the administrative console.	
server_generic_uuid	User cannot change.	Specifies the unique identifier for this server

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_classpath (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Java Virtual Machine , and, in the Classpath field, specify the classpath to be appended.	Specifies the classpath used by the servant region's JVM.
server_region_dynapplenv_jclparms (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Servant > Start command arguments , Specify the new parameters.	When dynamic applenv is being used (instead of the same content existing in static definition of WLM panels), this variable specifies the JCL parameters provided to the servant region when WLM starts this servant region.
server_region_dynapplenv_jclproc (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > servant > Start command , and then specify the new JCL procedure.	When dynamic applenv is used, this variable specifies the name of the JCL procedure for a servant region when WLM starts this servant region.
server_region_jvm_localrefs (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_jvm_localrefs property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_jvm_logfile (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_jvm_logfile property, and specify a different value.	Specifies the HFS file that JNI debug messages are written to.
server_region_jvm_properties_file (application server and deployment manager only)	User cannot change	File is dynamically created.
server_region_libpath (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Servant > Environment entries , and then specify a new value for the LIBPATH property.	Specifies the libpath for the servant region's JVM
server_region_recycle_count (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_recycle_count property, and specify a different value.	See Application server z/OS custom properties for additional information.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_http_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_http_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_https_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_https_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_iiop_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_iiop_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_mdb_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_mdb_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_sip_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_sip_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_sips_stalled_thread_dump_action (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_sips_stalled_thread_dump_action property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_stalled_thread_threshold_percent (application server and deployment manager only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_region_stalled_thread_threshold_percent property, and specify a different value.	See Application server z/OS custom properties for additional information.
server_region_thread_stack_size (application server and deployment manager only)		Should only be changed under the direction of IBM Support personnel.
server_region_use_java_g (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Java virtual machine , and select Debug Mode .	Indicates if the servant region's JVM should use the debug JVM (java_g). Should only be used under the direction of IBM support.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

server_region_workload_profile (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Container Settings, click Container Services > ORB Service .	
server_SMF_container_activity_enabled (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_container_activity_enabled property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_container_interval_enabled (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_container_interval_enabled property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_interval_length (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_interval_length property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request_activity_enabled	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_request_activity_enabled property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request_activity_enabled_CPU_detail	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_request_activity_enabled_CPU_detail property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_request_activity_enabled_security	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_request_activity_enabled_security property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

server_SMF_request_activity_enabled_timestamps	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_request_activity_enabled_timestamps property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_server_activity_enabled (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_server_activity_enabled property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_server_interval_enabled (application server and deployment manager only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Process definition > Control > Environment entries . Add the server_SMF_server_interval_enabled property and specify a different value.	See Using the WebSphere Application Server administrative console to enable properties for specific SMF record types for additional information.
server_SMF_web_container_activity_enabled	User cannot change.	No longer used.
server_SMF_web_container_interval_enabled	User cannot change.	No longer used.
serverRegionid	User cannot change.	No longer used.
server_specific_name	User cannot change.	
server_specific_short_name	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name , and specify a new value in the Short name field.	
server_specific_uuid	User cannot change.	Specifies the unique identifier for this server
server_start_wait_for_initialization_Timeout	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the server_start_wait_for_initialization_Timeout property, and specify a different value.	See Application server z/OS custom properties for additional information.
transaction_defaultTimeout (application server only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Container Settings, click Container Services > Transaction Service and specify a new value in the Transaction lifetime timeout field.	The maximum duration, in seconds, for transactions on this application server. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 120.

Table 4. Mapping internal variables reference in BBOM00011 messages to external WebSphere variable, custom property or administrative console fields (continued)

transaction_ maximumTimeout (application server only)	In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Container Settings, click Container Services > Transaction Service and specify a new value in the Maximum Transaction Timeout field.	The maximum duration, in seconds, that transactions propagated into the server or transactions started by BMT components from within the server will be allowed to execute. Any transaction that is not requested to complete before this timeout will be rolled back. Default is 300.
transaction_ recoveryTimeout (application server only)	In the administrative console, click Environment > WebSphere variables , select the appropriate node or cell from the list of available nodes and cells, and then click New . Add the transaction_recoveryTimeout property and specify a different value.	See Application server z/OS custom properties for additional information.
was_env_file	User cannot change.	File is dynamically created.
wlm_dynapplenv_ single_server (application server and deployment manager only)	For an application server, In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Server instance . For a deployment manager, in the administrative console, click System administration > Deployment manager . Then, under Server Infrastructure, click Java and process management > Server instance .	
wlm_maximumSRCount (application server and deployment manager only)	For an application server, In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Server instance . For a deployment manager, in the administrative console, click System administration > Deployment manager . Then, under Server Infrastructure, click Java and process management > Server instance .	
wlm_minimumSRCount (application server and deployment manager only)	For an application server, In the administrative console, click Servers > Server Types > WebSphere application servers > server_name . Then, under Server Infrastructure, click Java and process management > Server instance . For a deployment manager, in the administrative console, click System administration > Deployment manager . Then, under Server Infrastructure, click Java and process management > Server instance .	

About this task

After you determine which WebSphere variable, custom property or administrative console field needs to be updated, complete the following procedure to change the value of that setting.

1. Navigate to the appropriate administrative console panel. The "How to change indicated value" column describes how to navigate within the administrative console to the appropriate panel for changing a specific internal variable setting. For example, to change the setting of the wlm_maximumSRCount

variable for an application server, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Server instance**.

2. Update the variable, custom property, or administrative console field with the new value.
3. Click **Apply**, and then click **SAVE** to save your changes.

Environment entries collection

Use this page to view and manage arbitrary name-value pairs of data, where the name is a environment entry key and the value is a string value that can be used to set internal system configuration environment entries.

To view this page, in the administrative console click **Servers > Server Types > WebSphere application servers > *server_name***, and then under Server Infrastructure, click **Java and process management > Environment entries**.

Name

Specifies the name (or key) for the environment entry. The name is a string that is used to set an internal system configuration environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start your environment entry names with `was.` because this prefix is reserved for environment entries that are predefined for WebSphere Application Server.

Value

Specifies the value paired with the specified name.

Description

Provides information about the name-value pair.

Environment entries settings

Use this page to configure arbitrary name-value pairs of data, where the name is an environment entry key and the value is a string value that can be used to set internal system configuration environment entries. Defining a new environment entry enables you to configure a setting beyond that which is available in the administrative console.

To view this page, in the administrative console click **Servers > Server Types > WebSphere application servers > *server_name***. Under Server Infrastructure, click **Java and process management > Environment entries**. Then do one of the following:

- Click **New** to create a new environment entry.
- Click the name of an existing environment entry to change its settings,
- Select an existing environment entry and click **Delete** to delete that entry.

Name:

Specifies the name (or key) for the environment entry.

Each environment entry name must be unique. If the same name is used for multiple environment entries, the value specified for the first environment entry that has that name is used.

Do not start an environment entry name with `was.` because this prefix is reserved for environment entries that are predefined in WebSphere Application Server.

Data type String

Value:

Specifies the value paired with the specified name.

Data type String

Description:

Provides information about the name and value pair.

Data type String

Starting an application server

When you start an application server, a new server process starts. This new server process is based on the process definition settings of the current server configuration.

Before you begin

Before you start an application server, verify that all of the application required resources are available. You must also start all prerequisite subsystems.

If you want server components to dynamically start as they are needed by the installed applications, verify that the **Start components as needed** option is selected in the configuration settings for the application server before you start the application server. Selecting this option can improve startup time, and reduce the memory footprint of the application server. Starting components as they are needed is most effective if all of the applications that are deployed on the server are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

Note: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

About this task

The node agent for the node on which an application server resides must be running before you can start the application server.

This procedure for starting a server also typically applies to restarting a server. The one exception might be if a server fails and you want the recovery functions to complete their processing prior to new work being started on that server. In this situation, you must restart the server in recovery mode.

After you create a new application server definition, you can start, stop, or manage the new server using the administrative console, or you can use commands to complete these tasks for the new server.

After you start an application server, other processes might not immediately discover the running application server. Application servers are discovered by the node agent. However, node agents are discovered by the deployment manager. Even though node agents typically discover local application servers quickly, it might take a deployment manager up to 60 seconds to discover a node agent.

If you are using clusters, the **Initial State** property of the application server subcomponent is not intended to be used to control the state of individual servers in the cluster at the time the cluster is started. This property is intended only as a way to control the state of the subcomponent of a server. You should use

the Server options on the administrative console, or the startServer and stopServer command line commands to start and stop the individual servers of a cluster

There are several options available for starting an application server.

- You can use the administrative console:
 1. Click **Servers > Server Types > WebSphere application servers** to determine the node agent on which the application server that you are starting resides.
 2. Click **System Administration > Node agents**, and verify that the node agent is running.
If the node agent is not running, issue the startNode command. After a node agent completely stops running and remains stopped, you cannot remotely start the node agent from the Node Agents page. You must issue the startNode command to start the node agent on the node where it runs.
 3. Click **Servers > Server Types > WebSphere application servers** again and select the application server that you want to start.
 4. Click **Start**. You can view the status and any messages or logs to make sure the application server starts.
- You can issue a start command from the MVS console. A typical product run time includes two nodes:
 - Deployment manager node. This includes a location service daemon and a Deployment Manager with a controller and any number of servants
 - Application server node. This includes a location service daemon, a node agent with a controller, and an application server with a controller and any number of servants
 1. 1. To start a server, issue the following command all in uppercase and on a single line. (It is shown here on two lines for printing purposes.)

```
S procname, JOBNAME= server_shortcode,  
  ENV= cell_shortcode.node_shortcode.server_shortcode
```

where:

procname

Is the JCL procedure name in the proclib that is used to start the server.

server_shortcode

Is the short name of the server (or the step name used to start the procedure). This allows you to identify the address space that is running when you view it in the SDSF panels.

cell_shortcode.Node_shortcode.Server_shortcode

This element of the ENV parameter is a concatenation of the cell short name, the node short name, and the server short name.

For example:

```
S BB07ACR, JOBNAME= BB0S001,ENV=SY1.SY1.BB0S001
```

Note: If you are migrating from a previous version of the product and want to be able to restart the server in recovery mode, make sure that the ENV parameter for this procedure includes either the REC=N or the REC=Y element. If the ENV parameter includes the REC=N element, the setting is automatically changed to REC=Y if you specify -recovery when you restart the server. The REC=N element is automatically included on the ENV parameter if you did not migrate from a previous version of the product. The following example illustrates what your updated PROC statement might look like:

```
//BB07ACR PROC ENV=,PARMS=' ',REC=N,Z=BB07ACRZ
```

The following messages indicate that the controller is running:

```
$HASP100 BB07ACR ON STCINRDR  
$HASP373 BB07ACR STARTED  
BB000001I WEBSPHERE FOR Z/OS CONTROL PROCESS BB0DMNB/SY1/BB0C001/BB0S001  
  IS STARTING.  
IRR812I PROFILE BBO*.* (G) IN THE STARTED CLASS WAS USED TO START BB0S001S  
  WITH JOBNAME BB0S001S.
```

```

$HASP100 BBOS001S ON STCINRDR
$HASP373 BBOS001S STARTED
+BB000004I WEBSphere FOR Z/OS SERVANT PROCESS BBODMNB/SY1/BBOC001/BBOS001
  IS STARTING.
+BB000020I INITIALIZATION COMPLETE FOR WEBSphere FOR Z/OS SERVANT PROCESS
  BBOS001.
BB000019I INITIALIZATION COMPLETE FOR WEBSphere FOR Z/OS CONTROL PROCESS
  BBOS001.

```

2. The controller automatically issues a command, similar to the following command, to start the daemon.

```

S <dmn_proc>,JOBNAME=<dmn_jobname>,
  ENV=<cell_shortname.Node_shortname.daemon_instancename>

```

The following example shows you the messages that are displayed during daemon startup:

```

BB000001I WEBSphere FOR Z/OS CONTROL PROCESS BBODMNB/SY1/BBOC001/BBOS001
  IS STARTING.
IRR812I PROFILE BBO*.* (G) IN THE STARTED CLASS WAS USED TO START BBO7DMN
  WITH JOBNAME BBO7DMN.
$HASP100 BBO7DMN ON STCINRDR
$HASP373 BBO7DMN STARTED
BB000007I WEBSphere FOR Z/OS DAEMON BBODMNB/SY1/BBODMNB/SY1 IS STARTING.
IEC130I STEPLIB DD STATEMENT MISSING
ITT102I CTRACE WRITER BBOWTR IS ALREADY ACTIVE.
BB000215I PRODUCT 'WEBSphere FOR Z/OS' SUCCESSFULLY REGISTERED WITH IFAED SERVICE.
BB000015I INITIALIZATION COMPLETE FOR DAEMON SY1.

```

WLM issues a command, similar to the following command, to start servant address spaces:

```

S <Srv_Reg_Proc>,JOBNAME=<Server_shortname>,
  ENV=<Cell_shortname.Node_shortname.Server_shortname>

```

3. Determine if the daemon is up.

If the Daemon is up, use the administrative console to start the application server (see Step 1).

- You can issue a startServer command.

Results

The specified server starts. To verify that the server is in start state, in the administrative console, click **Servers > Server Types > WebSphere application servers**.

What to do next

After the server starts, deploy the applications that you want to run on this server.

If you need to start an application server with standard Java debugging enabled:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**.
2. Click the name of the application server with the processes that you want to trace and debug.
3. Under Server Infrastructure, click **Java and process management > Process definition**.
4. Select **Control**.
5. Select **Java virtual machine**.
6. On the Java virtual machine page, select the **Debug mode** option to enable the standard Java debugger. Set **Debug mode** arguments, if they are needed.
7. Click **OK**.
8. Save the changes to a configuration file
9. Stop the application server.
10. Start the application server again as previously described.

Restarting an application server in recovery mode

When an application server instance with active transactions in progress restarts after a failure, the transaction service uses recovery logs to complete the recovery process. These logs, which each transactional resource maintains, are used to rerun any InDoubt transactions and return the overall system to a self-consistent state.

Before you begin

If you are migrating from a previous version of the product, make sure that the REC parameter is included on the JCL procedure statement for the controller as either REC=N or REC=Y. If the JCL procedure does not specify either REC=N or REC=Y, the server does not restart in recovery mode even if you specify the -recovery option.

If the JCL procedures includes REC=N, the setting automatically changes to REC=Y if you specify -recovery when you restart the server. REC=N is automatically included on the JCL procedure if you did not migrate from a previous version of the product. Following is an example of what your updated PROC statement might look like:

```
//BB06ACR PROC PARMS= ' ',REC=N,Z=BB06ACRZ
```

About this task

When you restart an application server in recovery mode:

- Transactional resources complete the actions in their recovery logs and then shut down. This action frees up any resource locks that the application server held prior to the failure.
- During the recovery period, only the subset of application server functions that are necessary for transactional recovery to proceed are available.
- The application server does not accept new work during the recovery process.
- The application server shuts down when the recovery is complete.

This recovery process begins as soon as all of the necessary subsystems within the application server are available. If the application server is not restarted in recovery mode, the application server can start accepting new work as soon as the server is ready, which might occur before the recovery work has completed.

Normally, this process is not a problem. However, situations exist when your operating procedures might not be compatible with supporting recovery work and new work simultaneously. For example, you might have a high availability environment where the work handled by the application server that failed is immediately moved to another application server. This backup application server then exclusively processes the work from the application server that failed until recovery has completed on the failed application server and the two application servers can be re-synchronized. In this situation, you might want the failing application server to only perform its transactional recovery process and then shut down. You might not want this application server to start accepting new work while the recovery process is taking place.

To prevent the assignment of new work to an application server that is going through its transaction recovery process, restart the application server in recovery mode.

When you restart a failed application server, the node agent for the node on which the failed application server resides must be running before you can restart that application server.

If you want to be able restart an application server in recovery mode, you must perform the following steps before a failure occurs, and then restart the application server to enable your configuration changes:

- If the server is monitored by a node agent, you must clear the Automatic restart option for that server. Clearing this option prevents the node agent from automatically restarting the server in normal mode, before you have a chance to start it in recovery mode.
 1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***.
 2. In the Server Infrastructure section, click **Java and process management > Monitoring Policy**.
 3. Clear the Automatic restart option.
- If a catastrophic failure occurs that leaves InDoubt transactions, issue the **startServer *server_name* -recovery** command from the command line. This command restarts the server in recovery mode. You must issue the command from the *profile_root/bin* directory for the profile with which the server is associated.

Results

The application server restarts in recovery mode, performs transactional recovery, and shuts down. Any resource locks that the application server held prior to the failure are released.

What to do next

Configure the integrated high availability support for the transaction service subcomponent for peer recovery of transactions.

InFlight work and presumed abort mode:

Presumed abort mode is activated when a failure occurs before a distributed transaction starts to commit.

If you have a distributed transaction that spans several servers, transactional locks may be held by resource managers involved in that work. When a failure occurs before that distributed transaction has started to commit, the product and the resource managers go into presumed abort mode. In this mode, the resource managers rolls back the transaction.

- The effect of a server failure or communications failure will vary depending on which server is running the work at the time of failure.
- An OTS timeout may be required to rollback the subordinate branches of the distributed transaction tree.

Example: A common case of this is when you have a server B Web client that is driving a session bean in the same server. That session bean has executed work against entity beans in servers C and D. All of the servers are involved in the same distributed, global transaction. Suddenly, server B fails while the session bean is InFlight (meaning it hadn't started to commit yet). Servers C and D are waiting for more work or the start of the two-phase commit protocol, but, while in this state, the transactional locks may still be held by the resource managers. So, the server roles are as follows:

- Server A: Servlet/JavaServer Page executed
- Server B: Session bean accessed
- Server C: Entity bean accessed
- Server D: Entity bean accessed

After the timeout occurs, because the session bean is InFlight at the time of the failure, the product rolls back the transaction branch.

When local resource managers are involved, RRS ensures that they are called to perform presumed abort processing. When doing recovery, RRS works with the resource managers to ensure that the recovery is done properly. When a failure occurs while work is InFlight, RRS directs the resource managers involved in the local UR to rollback.

The product always assumes that there is recovery to do. Every time a server comes up, it does something different depending in which mode it is running:

- If the server is running in restart/recovery mode, the product checks to see whether there is any recovery required. If recovery is required, the product attempts to complete the recovery and either succeeds or terminates.
- If the server is running normally, the restart/recovery transaction does not have to complete before the server takes on new work. After the server determines what the restart work is, it begins to take in new work items. Processing of the restart/recovery transaction continues along with the processing of new work items.

IMS Connect considerations following server recovery:

After InDoubt and InFlight work completes, the product server shuts down. A new application server configured for that system is then started up to accept new work. Special considerations must be taken if you are using IMS Connect after recovering to an alternate system.

After server recovery is completed, IMS Connect starts, but is not usable without some manual intervention. On the current IMS Connect WTOR perform the following commands `nn,viewhws` followed by `nn,viewhwsnn,opens XXX` where `XXX` is the IMS subsystem name displayed in the result of the `nn,viewhws` query. The IMS datastore needs to reflect 'active' status, as is shown in the following example:

```
*17 HWSC0000I *IMS CONNECT READY*  IMSCONN
R 17,VIEWHWS
IEE600I REPLY TO 17 IS;VIEWHWS
HWSC0001I  HWS ID=IMSCONN      Racf=N
HWSC0001I      Maxsoc=100  Timeout=12000
HWSC0001I  Datastore=IMS      Status=ACTIVE
HWSC0001I      Group=IMSGROUP Member=IMSCONN
HWSC0001I      Target Member=IMSA
HWSC0001I  Port=9999      Status=ACTIVE
HWSC0001I      No active Clients
HWSC0001I  Port=LOCAL      Status=ACTIVE
HWSC0001I      No active Clients
```

After you complete the required manual intervention, IMS Connect is ready to handle new work on this server.

Detecting and handling problems with runtime components

You must monitor the status of runtime components to ensure that, once started, they remain operational as needed.

1. Regularly examine the status of runtime components.
 - Browse messages displayed under WebSphere Runtime Messages in the status area at the bottom of the console. The runtime event messages, marked with a red X, provide detailed information on event processing.
2. If an application stops running, examine the status of the application. If an application stops running when it should be operational, examine the status of the application on an Applications page and try restarting the application. If messages indicate that a server has stopped running, use the Application servers page to try restarting the server. If a cluster of servers stops running, use the Server Cluster page to try to restart the cluster. If the status of an application server is Unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.
3. If the runtime components do not restart, reexamine the messages and read information on problem determination to help you to restart the components.

Stopping an application server

Stopping an application server ends a server process based on the process definition settings in the current application server configuration.

Before you begin

Make sure you understand the impact of stopping a particular server has on your ability to handle work requests, especially if you need to maintain a highly available environment.

About this task

There are times you need to stop an application server. For example, you might have to apply service to an application running on that server, or you might want to change one of the application server's configuration setting. Use one of the following options when you need to stop an application server.

- For the z/OS and distributed platforms, except AIX, you can issue the **stopServer** command from the command line to stop a single server or the **stopManager** command to stop the deployment manager. You should not use the CANCEL appserver_proc_name command to stop a server. Every time a server is started, a new temp directory is created off of the servant process token, such as *profile_root/default/temp/node_name/server_name*. When the server is cleanly stopped, these temp directories are normally removed. However, if the server is frequently not stopped cleanly, which happens if you cancel rather than stop the server, these temp directories are not removed and the HFS used for these temp directories eventually becomes full. You can also prevent this storage problem from occurring if you precompile your JavaServer pages when you install an application or if you use the JspBatchCompiler function to precompile them before they are invoked.
- You can use the administrative console to stop an application server:
 1. In the administrative console, click **Servers > Server Types > WebSphere application servers**.
 2. Select the application server that you want stopped and click **Stop**.
 3. Confirm that you want to stop the application server.
 4. View the **Status** value and any messages or logs to see whether the application server stops.

Results

The specified server stops as soon as requests assigned to that server finish processing. To verify that the server is in stop state, in the administrative console, click **Servers > Server Types > WebSphere application servers**.

What to do next

If you experience any problems shutting down a server, see the *Troubleshooting and support* PDF.

Automatically rejecting work requests when no servant is available to process these requests

When a controller determines that a servant has terminated, the controller normally cleans up any other work requests that were dispatched in that servant. If that servant is the last servant, new work requests are placed in the request queue until a servant is available. Depending on how long it takes for a servant to become available, these requests might terminate because the time allowed to process a request has expired. To prevent this from happening, you can change the configuration settings for an application server to prevent the controller from accepting new requests.

About this task

Controllers receive application requests on a continual basis and dispatch them to a servant for processing. When a system level problem, such as a database error, occurs, request processing stops and requests pile up in the queues between the controller and the servants. During the time it takes for a servant to become available, requests continue to pile up in the queues until they start to time out. When a timeout occurs, the request that timed out is removed from the queue.

When a new servant is ready to start accepting requests, the next request in the queue might be so close to timing out that the dispatch process for the request cannot complete and the servant again is terminated by timeout processing. Again requests accumulate in the queue until another new servant is ready and potentially the same timeout problem occurs. When this problem keeps reoccurring, it is sometimes referred to as a *bouncing servant* problem. You can handle this problem in one of the following ways:

- You can configure the server to automatically detect a no-server situation and stop accepting requests until the minimum configured number of servants are ready to accept work. This is the simplest approach.
- You can create an automation routine to handle the problem if you are able to detect that you are having a system problem before servants are terminated because of timeouts. This automation routine can issue the `f server, pauselisteners` command to prevent requests from being accepted by this server. The routine must then detect when circumstances have changed and issue the `f server, resumelisteners` command when the detected system problem is resolved.
- You can configure the server to detect a no-server condition and stop accepting requests, and create the previously discussed automation routine. The automation routine must recognize the different processing that can take place because the server is configured to detect a no-server condition:
 - If the last servant terminates even though the `f server, pauselisteners` command is issued, the server starts to reject all requests and issues message BBOO0299I. The server automatically starts to accept requests when the minimum number of servants, for which the server is configured, are ready to accept work. It also issues message BBOO0300I to indicate that requests are again being processed. Therefore, the automation routine must be sensitive to the fact that the server might have resumed accepting requests upon detecting that the minimal number of servants are available.
 - If the `control_region_confirm_recovery_on_no_srs` custom property is specified for the server, the server issues WTOR message BBOO0297A after it detects that the minimal number of servants, for which the server is configured, are ready to process new requests. You must enter a response to this message before the server actually starts to accept work.
 - If the automation routine prevents the server from terminating the servants because of timeout processing, it must also recognize when it is safe for the server to resume taking requests and issue the `f server, resumelisteners` command at that point in time. The automation routine can be set up to determine whether or not it needs to issue the `f server, resumelisteners` command based on whether or not the message BBOO0299I is issued. This message indicates that the server ran out of servants and is rejecting requests. This approach is the most complex, but provides the most flexibility.

Complete the following steps if you want to configure the server to handle no-server conditions.

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and select the application server for which you want to automatically detect no-servant conditions.
2. Under Additional Properties, click **Custom properties > New**.
3. Specify `control_region_dreg_on_no_srs` in the Name field and 1 in the Value field. When this custom property is set to 1, the server rejects all requests targeted for dispatch when it detects that there are no servants ready to process the requests. Setting this property to 0 (zero) turns off this function.
4. Specify `control_region_confirm_recovery_on_no_srs` in the Name field and either 0 (zero) or 1 in the Value field. If you enter 0 in the Value field, the controller resumes taking requests as soon as the minimum number of servants are ready to receive requests. If you enter 1 in the Value field, the controller issues WTOR message BBOO0297A as soon as it detects that the minimum number of servants for which the server is configured are ready to accept work. The server waits until it receives a response to this message before it actually resumes taking requests.
5. Click **Review**, select **Synchronize changes with Nodes**, and then click **Save** to update the master repository with your changes.

Results

When a controller determines that a servant has terminated, and that servant is the last servant, the controller will not accept new work requests until the minimum number of servants are available to receive requests.

Converting a 7-character server short name to 8 characters

By default, the product assumes you are using a 7-character short name (JOBNAME) for your application servers and deployment managers. If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.

Before you begin

You should consider the following before performing this task:

- The Resource Recovery Services (RRS) log names are based on the server short name. When you change the server short name, you are changing the server's identity to the RRS. This means that the previously existing transaction and partner logs will be abandoned, or will not match the new name, and either of these situations will result in restart problems. To prevent this from happening, ensure that there are no outstanding RRS units of recovery (URs) for your server **before** changing its name. See *z/OS MVS Programming: Resource Recovery* for instructions on using the RRS panels to view information about URs.

Note that the only safe way to provide an 8-character short name for a server is to do so before it is initially started.

- Converting your 7-character server short name to 8 characters requires you to change the JOBNAME used by the servant's start command. This means that the System Authorization Facility (SAF) started class that previously matched this job may no longer match. Review your SAF STARTED class profile and, if necessary, define a new class.
- Because the JOBNAME appears as part of a start command's arguments, you need to review your COMMNDxx PARMLIB member, as well as any other form of automation you use that issues a start command to start a server.
- Review the start parameters of your Workload Management (WLM) static APPLENV definitions. These are the parameters that are used to start the servant process (server region). If you are using static APPLENVs, the start parm string used by the WLM for your server looks similar to JOBNAME=BBOS001S,ENV=... You will need to decide if you want to keep this JOBNAME or change to the new JOBNAME that you specify in the steps below. The original JOBNAME should be sufficient. This consideration does not apply if you are using WLM dynamic APPLENVs.
- Review and update the necessary Resource Access Control Facility (RACF) profiles to support these server short names. See the *Securing applications and their environment* PDF for more information.

About this task

To lengthen a 7-character server short name to 8 characters:

1. Change the 7-character server short name to the 8-character name you wish to use:
 - a. Navigate to the appropriate application server or deployment manager.
For an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**.
For a deployment manager, in the administrative console, click **System Administration > Deployment manager**.
 - b. In the **Short name** field, replace the 7-character name with the 8-character short name you wish to use.
 - c. Click **OK**.

2. Update the start command arguments for the servant to use the new 8-character name. If you are reconfiguring a node agent, you can skip this step because it does not have an associated servant process.
 - a. Navigate to the servant process.

For an application server, in the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Java and process management > Process definition > Servant**.

For a deployment manager, in the administrative console, click **System Administration > Deployment manager**, and then under Server Infrastructure, click **Java and process management > Process definition**.
 - b. In the startCommandArgs field, replace the 7-character name, designated by the JOBNAME argument, with the 8-character name you wish to use. Do not include the S character at the end of the JOBNAME. For example, JOBNAME=P5SVR1D,ENV=P5CELL.P5NODED.P5SVR1D
 - c. Click **OK**.
3. Click **Save** to save your configuration changes.

Changing time zone settings

In some application environments, it is important that application server components use the same time zone. You can use the administrative console to ensure that your application components use the correct time zone.

Before you begin

Determine the scope at which you want to set the time zone value. You can set the time zone value such that it applies for an entire cell, for an entire node, or only for a specific server.

Remember that time zone IDs should include an offset and, in almost all cases, a daylight saving time zone name for consistent results. For example, specify EST5EDT for Eastern Standard Time, Daylight Savings Time.

About this task

You can specify the Unix System Services (USS) TZ variable as a WebSphere variable to change your time zone setting.

To change the time zone setting for a single application server:

Complete one or more of the following actions to set appropriate time zone values for your environment.

- Set the time zone for all of your server processes.
 1. In the administrative console, click **Environment > WebSphere variables** .
 2. Select All scopes from the list of scope options.
 3. Set a value for the TZ variable.

If the TZ variable is included in the list of defined variables, click **TZ**, and then specify a new time zone value in the **Variable** field.

If the TZ variable is not included in the list of defined variables, click **New**, and then specify TZ in the **Name** field, and the appropriate time zone value in the **Value** field.

For example, if you specify TZ in the **Name** field, and EST5EDT in the **Value** field, Eastern Daylight Savings is used as the time zone setting for all of your server processes.
- Set the time zone for all of the server processes in a particular cell.
 1. In the administrative console, click **Environment > WebSphere variables** .
 2. Select the cell for which you want to set the time zone value from the list of scope options.

3. Set a value for the TZ variable.

If the TZ variable is included in the list of defined variables, click **TZ**, and specify a new time zone value in the **Value** field.

If the TZ variable is not included in the list of defined variables, click **New**, and then specify TZ in the **Name** field, and the appropriate time zone value in the **Value** field.

For example, if you specify TZ in the **Name** field, and EST5EDT in the **Value** field, Eastern Daylight Savings is used as the time zone setting for all of your server processes that are running in that cell.

- Set the time zone for all of the server processes in a particular node.

1. In the administrative console, click **Environment > WebSphere variables** .
2. Select the node for which you want to set the time zone value from the list of scope options.
3. Set a value for the TZ variable.

If the TZ variable is not included in the list of defined variables, click **New**, and then specify TZ in the **Name** field, and the appropriate time zone value in the **Value** field.

If the TZ variable is included in the list of defined variables, click **TZ**, and specify a new time zone value in the **Value** field.

For example, if you specify TZ in the **Name** field, and EST5EDT in the **Value** field, Eastern Daylight Savings is used as the time zone setting for all of your server processes that are running in that node.

- Set the time zone for a specific server.

1. In the administrative console, click **Environment > WebSphere variables** .
2. Select the server for which you want to set the time zone value from the list of scope options.
3. Set a value for the TZ variable.

If the TZ variable is included in the list of defined variables, click **TZ**, and then specify a new time zone value in the **Value** field.

If the TZ variable is not included in the list of defined variables, click **New**, and then specify TZ in the **Name** field, and the appropriate time zone value in the **Value** field.

For example, if you specify TZ in the **Name** field, and EST5EDT in the **Value** field, Eastern Daylight Savings is used as the time zone setting for all of your server processes.

- Set the time zone value for trace statements in SYSPRINT for a particular server or for the deployment manager.

1. Set a value for the ras_time_local variable.

If the ras_time_local variable is included in the list of defined variables, click **ras_time_local** , and then specify 1 in the **Value** field.

If the ras_time_local variable is not included in the list of defined variables, click **New**, and then specify ras_time_local in the **Name** field, and 1 in the **Value** field.

For example, if you specify ras_time_local in the **Name** field, and EST5EDT in the **Value** field, Eastern Daylight Savings is used as the time zone setting for SYSPRINT for the selected deployment manager or server.

- Click **Apply**, and then click **Save** to save your changes.
- Stop and restart all of the affected application server that were running when you made the time zone changes.

Results

Your new time zone setting are in affect for the designated servers.

Time zone IDs that can be specified for the user.timezone property

The following table lists the time zone IDs that you can specify for the user.timezone property.

- The **Time zone ID** column lists time zones, in boldface, and the locations within each time zone.

- The **Raw offset** column lists the difference, in hours and minutes, between Greenwich Mean Time (GMT) and the specified time zone.
- The **DST offset** column lists the offset, in minutes, for Daylight Savings Time (DST). If the field is blank, the time zone does not use DST.
- The **Display name** column lists the names of the time zones.
- The **QTIMZON variable** column only applies to the i5/OS operating system. The **QTIMZON variable** column lists the corresponding value for the QTIMZON system variable. If multiple values are specified in this column, either value is acceptable.

Note: The United States and Canada are making changes to the Daylight Saving Time start and end dates. The Technote Changes to Daylight Saving Time will affect IBM WebSphere Application Server and its associated Operating Systems, that is available on the Support Web site, provides the latest information on service updates that are being made to support these changes.

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Etc/GMT+12	-12 : 00		GMT-12:00	
Etc/GMT+11	-11 : 00		GMT-11:00	
MIT	-11 : 00		West Samoa Time	
Pacific/Apia	-11 : 00		West Samoa Time	QN1100UTCS
Pacific/Midway	-11 : 00		Samoa Standard Time	
Pacific/Niue	-11 : 00		Niue Time	
Pacific/Pago_Pago	-11 : 00		Samoa Standard Time	
Pacific/Samoa	-11 : 00		Samoa Standard Time	
US/Samoa	-11 : 00		Samoa Standard Time	
America/Adak	-10 : 00	60	Hawaii-Aleutian Standard Time	QN1000HAST
America/Atka	-10 : 00	60	Hawaii-Aleutian Standard Time	
Etc/GMT+10	-10 : 00		GMT-10:00	
HST	-10 : 00		Hawaii Standard Time	
Pacific/Fakaofu	-10 : 00		Tokelau Time	
Pacific/Honolulu	-10 : 00		Hawaii Standard Time	QN1000UTCS
Pacific/Johnston	-10 : 00		Hawaii Standard Time	
Pacific/Rarotonga	-10 : 00		Cook Is. Time	
Pacific/Tahiti	-10 : 00		Tahiti Time	
SystemV/HST10	-10 : 00		Hawaii Standard Time	
US/Aleutian	-10 : 00	60	Hawaii-Aleutian Standard Time	
US/Hawaii	-10 : 00		Hawaii Standard Time	
Pacific/Marquesas	-9 : 30		Marquesas Time	
AST	-9 : 00	60	Alaska Standard Time	QN0900AST
America/Anchorage	-9 : 00	60	Alaska Standard Time	
America/Juneau	-9 : 00	60	Alaska Standard Time	
America/Nome	-9 : 00	60	Alaska Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
America/Yakutat	-9 : 00	60	Alaska Standard Time	
Etc/GMT+9	-9 : 00		GMT-09:00	
Pacific/Gambier	-9 : 00		Gambier Time	QN0900UTCS
SystemV/YST9	-9 : 00	60	Alaska Standard Time	
US/Alaska	-9 : 00	60	Alaska Standard Time	
America/Dawson	-8 : 00	60	Pacific Standard Time	
America/Ensenada	-8 : 00	60	Pacific Standard Time	
America/Los_Angeles	-8 : 00	60	Pacific Standard Time	
America/Tijuana	-8 : 00	60	Pacific Standard Time	
America/Vancouver	-8 : 00	60	Pacific Standard Time	
America/Whitehorse	-8 : 00	60	Pacific Standard Time	
Canada/Pacific	-8 : 00	60	Pacific Standard Time	
Canada/Yukon	-8 : 00	60	Pacific Standard Time	
Etc/GMT+8	-8 : 00		GMT-08:00	
Mexico/BajaNorte	-8 : 00	60	Pacific Standard Time	
PST	-8 : 00	60	Pacific Standard Time	QN0800PST, QN0800U
PST8PDT	-8 : 00	60	Pacific Standard Time	
Pacific/Pitcairn	-8 : 00		Pitcairn Standard Time	QN0800UTCS
SystemV/PST8	-8 : 00		Pitcairn Standard Time	
SystemV/PST8PDT	-8 : 00	60	Pacific Standard Time	
US/Pacific	-8 : 00	60	Pacific Standard Time	
US/Pacific-New	-8 : 00	60	Pacific Standard Time	
America/Boise	-7 : 00	60	Mountain Standard Time	
America/Cambridge_Bay	-7 : 00	60	Mountain Standard Time	
America/Chihuahua	-7 : 00	60	Mountain Standard Time	
America/Dawson_Creek	-7 : 00		Mountain Standard Time	
America/Denver	-7 : 00	60	Mountain Standard Time	
America/Edmonton	-7 : 00	60	Mountain Standard Time	
America/Hermosillo	-7 : 00		Mountain Standard Time	
America/Inuvik	-7 : 00	60	Mountain Standard Time	
America/Mazatlan	-7 : 00	60	Mountain Standard Time	
America/Phoenix	-7 : 00		Mountain Standard Time	QN0700MST2, QN0700UTCS
America/Shiprock	-7 : 00	60	Mountain Standard Time	
America/Yellowknife	-7 : 00	60	Mountain Standard Time	
Canada/Mountain	-7 : 00	60	Mountain Standard Time	
Etc/GMT+7	-7 : 00		GMT-07:00	
MST	-7 : 00	60	Mountain Standard Time	QN0700MST, QN0700T

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
MST7MDT	-7 : 00	60	Mountain Standard Time	
Mexico/BajaSur	-7 : 00	60	Mountain Standard Time	
Navajo	-7 : 00	60	Mountain Standard Time	
PNT	-7 : 00	60	Mountain Standard Time	
SystemV/MST7	-7 : 00		Mountain Standard Time	
SystemV/MST7MDT	-7 : 00	60	Mountain Standard Time	
UA/Arizona	-7 : 00		Mountain Standard Time	
US/Mountain	-7 : 00	60	Mountain Standard Time	
America/Belize	-6 : 00		Central Standard Time	
America/Cancun	-6 : 00	60	Central Standard Time	
America/Chicago	-6 : 00	60	Central Standard Time	
America/Costa_Rica	-6 : 00		Central Standard Time	QN0600UTCS
America/El_Salvador	-6 : 00		Central Standard Time	
America/Guatemala	-6 : 00		Central Standard Time	
America/Managua	-6 : 00		Central Standard Time	
America/Menominee	-6 : 00	60	Central Standard Time	
America/Merida	-6 : 00	60	Central Standard Time	
America/Mexico_City	-6 : 00	60	Central Standard Time	
America/Monterrey	-6 : 00	60	Central Standard Time	
America/North_Dakota/Center	-6 : 00	60	Central Standard Time	
America/Rainy_River	-6 : 00	60	Central Standard Time	
America/Rankin_Inlet	-6 : 00	60	Central Standard Time	
America/Regina	-6 : 00		Central Standard Time	
America/Swift_Current	-6 : 00		Central Standard Time	
America/Tegucigalpa	-6 : 00		Central Standard Time	
America/Winnipeg	-6 : 00	60	Central Standard Time	
CST	-6 : 00	60	Central Standard Time	QN0600CST, QN600S
CST6CDT	-6 : 00	60	Central Standard Time	
Canada/Central	-6 : 00	60	Central Standard Time	
Canada/East-Saskatchewan	-6 : 00		Central Standard Time	
Canada/Saskatchewan	-6 : 00		Central Standard Time	
Chile/EasterIsland	-6 : 00	60	Easter Is.Time	
Etc/GMT+6	-6 : 00		GMT-06:00	
Mexico/General	-6 : 00	60	Central Standard Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
Pacific/Easter	-6 : 00	60	Easter Is. Time	
Pacific/Galapagos	-6 : 00		Galapagos Time	
SystemV/CST6	-6 : 00		Central Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
SystemV/CST6CDT	-6 : 00	60	Central Standard Time	
US/Central	-6 : 00	60	Central Standard Time	
America/Bogota	-5 : 00		Colombia Time	
America/Cayman	-5 : 00		Eastern Standard Time	
America/Detroit	-5 : 00	60	Eastern Standard Time	
America/Eirunepe	-5 : 00		Acre Time	
America/Fort_Wayne	-5 : 00		Eastern Standard Time	
America/Grand_Turk	-5 : 00	60	Eastern Standard Time	
America/Guayaquil	-5 : 00		Ecuador Time	
America/Havana	-5 : 00	60	Central Standard Time	
America/Indiana/Indianapolis	-5 : 00		Eastern Standard Time	
America/Indiana/Knox	-5 : 00		Eastern Standard Time	
America/Indiana/Marengo	-5 : 00		Eastern Standard Time	
America/Indiana/Vevay	-5 : 00		Eastern Standard Time	
America/Indianapolis	-5 : 00		Eastern Standard Time	QN0500UTCS
America/Iqaluit	-5 : 00	60	Eastern Standard Time	
America/Jamaica	-5 : 00		Eastern Standard Time	
America/Kentucky/Louisville	-5 : 00	60	Eastern Standard Time	
America/Kentucky/Monticello	-5 : 00	60	Eastern Standard Time	
America/Knox_IN	-5 : 00		Eastern Standard Time	
America/Lima	-5 : 00		Peru Time	
America/Louisville	-5 : 00	60	Eastern Standard Time	
America/Montreal	-5 : 00	60	Eastern Standard Time	
America/Nassau	-5 : 00	60	Eastern Standard Time	
America/New_York	-5 : 00	60	Eastern Standard Time	
America/Nipigon	-5 : 00	60	Eastern Standard Time	
America/Panama	-5 : 00		Eastern Standard Time	
America/Pangnirtung	-5 : 00	60	Eastern Standard Time	
America/Port-au-Prince	-5 : 00		Eastern Standard Time	
America/Porto_Acre	-5 : 00		Acre Time	
America/Rio_Branco	-5 : 00		Acre Time	
America/Thunder_Bay	-5 : 00	60	Eastern Standard Time	
Brazil/Acre	-5 : 00		Acre Time	
Canada/Eastern	-5 : 00	60	Eastern Standard Time	
Cuba	-5 : 00	60	Central Standard Time	
EST	-5 : 00	60	Eastern Standard Time	QN0500EST
EST5EDT	-5 : 00	60	Eastern Standard Time	
Etc/GMT+5	-5 : 00		GMT-05:00	
IET	-5 : 00		Eastern Standard Time	QN0500EST2

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Jamaica	-5 : 00		Eastern Standard Time	
SystemV/EST5	-5 : 00		Eastern Standard Time	
SystemV/EST5EDT	-5 : 00	60	Eastern Standard Time	
US/East-Indiana	-5 : 00		Eastern Standard Time	
US/Eastern	-5 : 00	60	Eastern Standard Time	
US/Indiana-Starke	-5 : 00		Eastern Standard Time	
US/Michigan	-5 : 00	60	Eastern Standard Time	
America/Anguilla	-4 : 00		Atlantic Standard Time	
America/Antigua	-4 : 00		Atlantic Standard Time	
America/Aruba	-4 : 00		Atlantic Standard Time	
America/Asuncion	-4 : 00	60	Paraguay Time	
America/Barbados	-4 : 00		Atlantic Standard Time	
America/Boa_Vista	-4 : 00		Amazon Standard Time	
America/Caracas	-4 : 00		Venezuela Time	QN0400UTC2
America/Cuiaba	-4 : 00	60	Amazon Standard Time	
America/Curacao	-4 : 00		Atlantic Standard Time	
America/Dominica	-4 : 00		Atlantic Standard Time	
America/Glace_Bay	-4 : 00	60	Atlantic Standard Time	
America/Goose_Bay	-4 : 00	60	Atlantic Standard Time	
America/Grenada	-4 : 00		Atlantic Standard Time	
America/Guadeloupe	-4 : 00		Atlantic Standard Time	
America/Guyana	-4 : 00		Guyana Time	
America/Halifax	-4 : 00	60	Atlantic Standard Time	
America/La_Paz	-4 : 00		Bolivia Time	
America/Manaus	-4 : 00		Amazon Standard Time	
America/Martinique	-4 : 00		Atlantic Standard Time	
America/Montserrat	-4 : 00		Atlantic Standard Time	
America/Port_of_Spain	-4 : 00		Atlantic Standard Time	
America/Porto_Velho	-4 : 00		Amazon Standard Time	
America/Puerto_Rico	-4 : 00		Atlantic Standard Time	QN0400UTCS
America/Santiago	-4 : 00	60	Chile Time	
America/Santo_Domingo	-4 : 00		Atlantic Standard Time	
America/St_Kitts	-4 : 00		Atlantic Standard Time	
America/St_Lucia	-4 : 00		Atlantic Standard Time	
America/St_Thomas	-4 : 00		Atlantic Standard Time	
America/St_Vincent	-4 : 00		Atlantic Standard Time	
America/Thule	-4 : 00	60	Atlantic Standard Time	
America/Tortola	-4 : 00		Atlantic Standard Time	
America/Virgin	-4 : 00		Atlantic Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Antarctica/Palmer	-4 : 00	60	Chile Time	
Atlantic/Bermuda	-4 : 00	60	Atlantic Standard Time	QN0400AST
Atlantic/Stanley	-4 : 00	60	Falkland Is. Time	
Brazil/West	-4 : 00		Amazon Standard Time	
Canada/Atlantic	-4 : 00	60	Atlantic Standard Time	
Chile/Continental	-4 : 00	60	Chile Time	
Etc/GMT+4	-4 : 00		GMT-04:00	
PRT	-4 : 00		Atlantic Standard Time	
SystemV/AST4	-4 : 00		Atlantic Standard Time	
SystemV/AST4ADT	-4 : 00	60	Atlantic Standard Time	
America/St_Johns	-3 : 30	60	Newfoundland Standard Time	
CNT	-3 : 30	60	Newfoundland Standard Time	QN0330NST
Canada/Newfoundland	-3 : 30	60	Newfoundland Standard Time	
AGT	-3 : 00		Argentine Time	
America/Araguaina	-3 : 00	60	Brazil Time	
America/Belem	-3 : 00		Brazil Time	
America/Buenos_Aires	-3 : 00		Argentine Time	QN0300UTCS
America/Catamarca	-3 : 00		Argentine Time	
America/Cayenne	-3 : 00		French Guiana Time	
America/Cordoba	-3 : 00		Argentine Time	
America/Fortaleza	-3 : 00		Brazil Time	
America/Godthab	-3 : 00	60	Western Greenland Time	
America/Jujuy	-3 : 00		Argentine Time	
America/Maceio	-3 : 00		Brazil Time	
America/Mendoza	-3 : 00		Argentine Time	
America/Miquelon	-3 : 00	60	Pierre & Miquelon Standard Time	
America/Montevideo	-3 : 00		Uruguay Time	
America/Paramaribo	-3 : 00		Suriname Time	
America/Recife	-3 : 00		Brazil Time	
America/Rosario	-3 : 00		Argentine Time	
America/Sao_Paulo	-3 : 00	60	Brazil Time	
Antarctica/Rothera	-3 : 00		Rothera Time	
BET	-3 : 00	60	Brazil Time	QN0300UTC2
Brazil/East	-3 : 00	60	Brazil Time	
Etc/GMT+3	-3 : 00		GMT-03:00	
America/Noronha	-2 : 00		Fernando de Noronha Time	QN0200UTCS

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Atlantic/South_Georgia	-2 : 00		South Georgia Standard Time	
Brazil/DeNoronha	-2 : 00		Fernando de Noronha Time	
Etc/GMT+2	-2 : 00		GMT-02:00	
America/Scoresbysund	-1 : 00	60	Eastern Greenland Time	
Atlantic/Azores	-1 : 00	60	Azores Time	
Atlantic/Cape_Verde	-1 : 00		Cape Verde Time	QN0100UTCS
Etc/GMT+1	-1 : 00		GMT-01:00	
Africa/Abidjan	0 : 00		Greenwich Mean Time	
Africa/Accra	0 : 00		Greenwich Mean Time	
Africa/Bamako	0 : 00		Greenwich Mean Time	
Africa/Banjul	0 : 00		Greenwich Mean Time	
Africa/Bissau	0 : 00		Greenwich Mean Time	
Africa/Casablanca	0 : 00		Western European Time	
Africa/Conakry	0 : 00		Greenwich Mean Time	
Africa/Dakar	0 : 00		Greenwich Mean Time	
Africa/El_Aaiun	0 : 00		Western European Time	
Africa/Freetown	0 : 00		Greenwich Mean Time	
Africa/Lome	0 : 00		Greenwich Mean Time	
Africa/Monrovia	0 : 00		Greenwich Mean Time	
Africa/Nouakchott	0 : 00		Greenwich Mean Time	
Africa/Ouagadougou	0 : 00		Greenwich Mean Time	
Africa/Sao_Tome	0 : 00		Greenwich Mean Time	
Africa/Timbuktu	0 : 00		Greenwich Mean Time	
America/Danmarkshavn	0 : 00		Greenwich Mean Time	
Atlantic/Canary	0 : 00	60	Western European Time	
Atlantic/Faeroe	0 : 00	60	Western European Time	
Atlantic/Madeira	0 : 00	60	Western European Time	
Atlantic/Reykjavik	0 : 00		Greenwich Mean Time	
Atlantic/St_Helena	0 : 00		Greenwich Mean Time	
Eire	0 : 00	60	Greenwich Mean Time	
Etc/GMT	0 : 00		GMT+00:00	
Etc/GMT+0	0 : 00		GMT+00:00	
Etc/GMT-0	0 : 00		GMT+00:00	
Etc/GMT0	0 : 00		GMT+00:00	
Etc/Greenwich	0 : 00		Greenwich Mean Time	
Etc/UCT	0 : 00		Coordinated Universal Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Etc/UTC	0 : 00		Coordinated Universal Time	
Etc/Universal	0 : 00		Coordinated Universal Time	
Etc/Zulu	0 : 00		Coordinated Universal Time	
Europe/Belfast	0 : 00	60	Greenwich Mean Time	
Europe/Dublin	0 : 00	60	Greenwich Mean Time	
Europe/Lisbon	0 : 00	60	Western European Time	
Europe/London	0 : 00	60	Greenwich Mean Time	Q0000GMT2
GB	0 : 00	60	Greenwich Mean Time	
GB-Eire	0 : 00	60	Greenwich Mean Time	
GMT	0 : 00		Greenwich Mean Time	Q0000GMT
GMT0	0 : 00		GMT+00:00	
Greenwich	0 : 00		Greenwich Mean Time	
Iceland	0 : 00		Greenwich Mean Time	
Portugal	0 : 00	60	Western European Time	
UCT	0 : 00		Coordinated Universal Time	
UTC	0 : 00		Coordinated Universal Time	Q0000UTC
Universal	0 : 00		Coordinated Universal Time	
WET	0 : 00	60	Western European Time	
Zulu	0 : 00		Coordinated Universal Time	
Africa/Algiers	1 : 00		Central European Time	QP0100CET, QP0100UTCS
Africa/Bangui	1 : 00		Western African Time	
Africa/Brazzaville	1 : 00		Western African Time	
Africa/Ceuta	1 : 00	60	Central European Time	
Africa/Douala	1 : 00		Western African Time	
Africa/Kinshasa	1 : 00		Western African Time	
Africa/Lagos	1 : 00		Western African Time	
Africa/Libreville	1 : 00		Western African Time	
Africa/Luanda	1 : 00		Western African Time	
Africa/Malabo	1 : 00		Western African Time	
Africa/Ndjamena	1 : 00		Western African Time	
Africa/Niamey	1 : 00		Western African Time	
Africa/Porto-Novo	1 : 00		Western African Time	
Africa/Tunis	1 : 00		Central European Time	
Africa/Windhoek	1 : 00	60	Western African Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Arctic/Longyearbyen	1 : 00	60	Central European Time	
Atlantic/Jan_Mayen	1 : 00	60	Eastern Greenland Time	
CET	1 : 00	60	Central European Time	
ECT	1 : 00	60	Central European Time	QP0100CET3
Etc/GMT-1	1 : 00		GMT+01:00	
Europe/Amsterdam	1 : 00	60	Central European Time	
Europe/Andorra	1 : 00	60	Central European Time	
Europe/Belgrade	1 : 00	60	Central European Time	
Europe/Berlin	1 : 00	60	Central European Time	
Europe/Bratislava	1 : 00	60	Central European Time	
Europe/Brussels	1 : 00	60	Central European Time	
Europe/Budapest	1 : 00	60	Central European Time	
Europe/Copenhagen	1 : 00	60	Central European Time	
Europe/Gibraltar	1 : 00	60	Central European Time	
Europe/Ljubljana	1 : 00	60	Central European Time	
Europe/Luxembourg	1 : 00	60	Central European Time	
Europe/Madrid	1 : 00	60	Central European Time	
Europe/Malta	1 : 00	60	Central European Time	
Europe/Monaco	1 : 00	60	Central European Time	
Europe/Oslo	1 : 00	60	Central European Time	
Europe/Paris	1 : 00	60	Central European Time	
Europe/Prague	1 : 00	60	Central European Time	
Europe/Rome	1 : 00	60	Central European Time	
Europe/San_Marino	1 : 00	60	Central European Time	
Europe/Sarajevo	1 : 00	60	Central European Time	
Europe/Skopje	1 : 00	60	Central European Time	
Europe/Stockholm	1 : 00	60	Central European Time	
Europe/Tirane	1 : 00	60	Central European Time	
Europe/Vaduz	1 : 00	60	Central European Time	
Europe/Vatican	1 : 00	60	Central European Time	
Europe/Vienna	1 : 00	60	Central European Time	
Europe/Warsaw	1 : 00	60	Central European Time	
Europe/Zagreb	1 : 00	60	Central European Time	
Europe/Zurich	1 : 00	60	Central European Time	QP0100CET2
MET	1 : 00	60	Middle Europe Time	
Poland	1 : 00	60	Central European Time	
ART	2 : 00	60	Eastern European Time	
Africa/Blantyre	2 : 00		Central African Time	
Africa/Bujumbura	2 : 00		Central African Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Africa/Cairo	2 : 00	60	Eastern European Time	
Africa/Gaborone	2 : 00		Central African Time	
Africa/Harare	2 : 00		Central African Time	
Africa/Johannesburg	2 : 00		South Africa Standard Time	QP0200SAST
Africa/Kigali	2 : 00		Central African Time	
Africa/Lubumbashi	2 : 00		Central African Time	
Africa/Lusaka	2 : 00		Central African Time	
Africa/Maputo	2 : 00		Central African Time	
Africa/Maseru	2 : 00		South Africa Standard Time	
Africa/Mbabane	2 : 00		South Africa Standard Time	
Africa/Tripoli	2 : 00		Eastern European Time	
Asia/Amman	2 : 00	60	Eastern European Time	
Asia/Beirut	2 : 00	60	Eastern European Time	
Asia/Damascus	2 : 00	60	Eastern European Time	
Asia/Gaza	2 : 00	60	Eastern European Time	
Asia/Istanbul	2 : 00	60	Eastern European Time	
Asia/Jerusalem	2 : 00	60	Israel Standard Time	
Asia/Nicosia	2 : 00	60	Eastern European Time	
Asia/Tel_Aviv	2 : 00	60	Israel Standard Time	
CAT	2 : 00		Central African Time	
EET	2 : 00	60	Eastern European Time	QP0200EET
Egypt	2 : 00	60	Eastern European Time	
Etc/GMT-2	2 : 00		GMT+02:00	
Europe/Athens	2 : 00	60	Eastern European Time	
Europe/Bucharest	2 : 00	60	Eastern European Time	
Europe/Chisinau	2 : 00	60	Eastern European Time	
Europe/Helsinki	2 : 00	60	Eastern European Time	
Europe/Istanbul	2 : 00	60	Eastern European Time	
Europe/Kaliningrad	2 : 00	60	Eastern European Time	
Europe/Kiev	2 : 00	60	Eastern European Time	
Europe/Minsk	2 : 00	60	Eastern European Time	
Europe/Nicosia	2 : 00	60	Eastern European Time	
Europe/Riga	2 : 00	60	Eastern European Time	
Europe/Simferopol	2 : 00	60	Eastern European Time	
Europe/Sofia	2 : 00	60	Eastern European Time	
Europe/Tallinn	2 : 00	60	Eastern European Time	QP0200EET2, QP0200UTCS

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Europe/Tiraspol	2 : 00	60	Eastern European Time	
Europe/Uzhgorod	2 : 00	60	Eastern European Time	
Europe/Vilnius	2 : 00	60	Eastern European Time	
Europe/Zaporozhye	2 : 00	60	Eastern European Time	
Israel	2 : 00	60	Israel Standard Time	
Libya	2 : 00		Eastern European Time	
Turkey	2 : 00	60	Eastern European Time	
Africa/Addis_Ababa	3 : 00		Eastern African Time	QP0300UTCS
Africa/Asmera	3 : 00		Eastern African Time	
Africa/Dar_es_Salaam	3 : 00		Eastern African Time	
Africa/Djibouti	3 : 00		Eastern African Time	
Africa/Kampala	3 : 00		Eastern African Time	
Africa/Khartoum	3 : 00		Eastern African Time	
Africa/Mogadishu	3 : 00		Eastern African Time	
Africa/Nairobi	3 : 00		Eastern African Time	
Antarctica/Syowa	3 : 00		Syowa Time	
Asia/Aden	3 : 00		Arabia Standard Time	
Asia/Baghdad	3 : 00	60	Arabia Standard Time	
Asia/Bahrain	3 : 00		Arabia Standard Time	
Asia/Kuwait	3 : 00		Arabia Standard Time	
Asia/Qatar	3 : 00		Arabia Standard Time	
Asia/Riyadh	3 : 00		Arabia Standard Time	
EAT	3 : 00		Eastern African Time	
Etc/GMT-3	3 : 00		GMT+03:00	
Europe/Moscow	3 : 00	60	Moscow Standard Time	
Indian/Antananarivo	3 : 00		Eastern African Time	
Indian/Comoro	3 : 00		Eastern African Time	
Indian/Mayotte	3 : 00		Eastern African Time	
W-SU	3 : 00	60	Moscow Standard Time	
Asia/Riyadh87	3 : 07		GMT+03:07	
Asia/Riyadh88	3 : 07		GMT+03:07	
Asia/Riyadh89	3 : 07		GMT+03:07	
Mideast/Riyadh87	3 : 07		GMT+03:07	
Mideast/Riyadh88	3 : 07		GMT+03:07	
Mideast/Riyadh89	3 : 07		GMT+03:07	
Asia/Tehran	3 : 30	60	Iran Standard Time	
Iran	3 : 30	60	Iran Standard Time	
Asia/Aqtau	4 : 00	60	Aqtau Time	QP0400UTC2
Asia/Baku	4 : 00	60	Azerbaijan Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Asia/Dubai	4 : 00		Gulf Standard Time	QP0400UTCS
Asia/Muscat	4 : 00		Gulf Standard Time	
Asia/Oral	4 : 00	60	Oral Time	
Asia/Tbilisi	4 : 00	60	Georgia Time	
Asia/Yerevan	4 : 00	60	Armenia Time	
Etc/GMT-4	4 : 00		GMT+04:00	
Europe/Samara	4 : 00	60	Samara Time	
Indian/Mahe	4 : 00		Seychelles Time	
Indian/Mauritius	4 : 00		Mauritius Time	
Indian/Reunion	4 : 00		Reunion Time	
NET	4 : 00	60	Armenia Time	
Asia/Kabul	4 : 30		Afghanistan Time	
Asia/Aqtobe	5 : 00	60	Aqtobe Time	QP0500UTC2
Asia/Ashgabat	5 : 00		Turkmenistan Time	
Asia/Ashkhabad	5 : 00		Turkmenistan Time	
Asia/Bishkek	5 : 00	60	Kirgizstan Time	
Asia/Dushanbe	5 : 00		Tajikistan Time	
Asia/Karachi	5 : 00		Pakistan Time	QP0500UTCS
Asia/Samarkand	5 : 00		Turkmenistan Time	
Asia/Tashkent	5 : 00		Uzbekistan Time	
Asia/Yekaterinburg	5 : 00	60	Yekaterinburg Time	
Etc/GMT-5	5 : 00		GMT+05:00	
Indian/Kerguelen	5 : 00		French Southern & Antarctic Lands Time	
Indian/Maldives	5 : 00		Maldives Time	
PLT	5 : 00		Pakistan Time	
Asia/Calcutta	5 : 30		India Standard Time	
IST	5 : 30		India Standard Time	QP0530IST
Asia/Katmandu	5 : 45		Nepal Time	
Antarctica/Mawson	6 : 00		Mawson Time	
Antarctica/Vostok	6 : 00		Vostok Time	
Asia/Almaty	6 : 00	60	Alma-Ata Time	QP0600UTC2
Asia/Colombo	6 : 00		Sri Lanka Time	
Asia/Dacca	6 : 00		Bangladesh Time	
Asia/Dhaka	6 : 00		Bangladesh Time	QP0600UTCS
Asia/Novosibirsk	6 : 00	60	Novosibirsk Time	
Asia/Omsk	6 : 00	60	Omsk Time	
Asia/Qyzylorda	6 : 00	60	Qyzylorda Time	
Asia/Thimbu	6 : 00		Bhutan Time	
Asia/Thimphu	6 : 00		Bhutan Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
BST	6 : 00		Bangladesh Time	
Etc/GMT-6	6 : 00		GMT+06:00	
Indian/Chagos	6 : 00		Indian Ocean Territory Time	
Asia/Rangoon	6 : 30		Myanmar Time	
Indian/Cocos	6 : 30		Cocos Islands Time	
Antarctica/Davis	7 : 00		Davis Time	
Asia/Bangkok	7 : 00		Indochina Time	
Asia/Hovd	7 : 00		Hovd Time	
Asia/Jakarta	7 : 00		West Indonesia Time	QP0700WIB
Asia/Krasnoyarsk	7 : 00	60	Krasnoyarsk Time	
Asia/Phnom_Penh	7 : 00		Indochina Time	
Asia/Pontianak	7 : 00		West Indonesia Time	
Asia/Saigon	7 : 00		Indochina Time	QP0700UTCS
Asia/Vientiane	7 : 00		Indochina Time	
Etc/GMT-7	7 : 00		GMT+07:00	
Indian/Christmas	7 : 00		Christmas Island Time	
VST	7 : 00		Indochina Time	
Antarctica/Casey	8 : 00		Western Standard Time (Australia)	
Asia/Brunei	8 : 00		Brunei Time	
Asia/Chongqing	8 : 00		China Standard Time	
Asia/Chungking	8 : 00		China Standard Time	
Asia/Harbin	8 : 00		China Standard Time	
Asia/Hong_Kong	8 : 00		Hong Kong Time	QP0800JIST, QP0800UTCS
Asia/Irkutsk	8 : 00	60	Irkutsk Time	
Asia/Kashgar	8 : 00		China Standard Time	
Asia/Kuala_Lumpur	8 : 00		Malaysia Time	
Asia/Kuching	8 : 00		Malaysia Time	
Asia/Macao	8 : 00		China Standard Time	
Asia/Macau	8 : 00		China Standard Time	
Asia/Makassar	8 : 00		Central Indonesia Time	
Asia/Manila	8 : 00		Philippines Time	
Asia/Shanghai	8 : 00		China Standard Time	
Asia/Singapore	8 : 00		Singapore Time	
Asia/Taipei	8 : 00		China Standard Time	
Asia/Ujung_Pandang	8 : 00		Central Indonesia Time	QP0800WITA
Asia/Ulaanbaatar	8 : 00		Ulaanbaatar Time	
Asia/Ulan_Bator	8 : 00		Ulaanbaatar Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Asia/Urumqi	8 : 00		China Standard Time	
Australia/Perth	8 : 00		Western Standard Time (Australia)	QP0800AWST
Australia/West	8 : 00		Western Standard Time (Australia)	
CTT	8 : 00		China Standard Time	QP0800BST
Etc/GMT-8	8 : 00		GMT+08:00	
Hongkong	8 : 00		Hong Kong Time	
PRC	8 : 00		China Standard Time	
Singapore	8 : 00		Singapore Time	
Asia/Choibalsan	9 : 00		Choibalsan Time	
Asia/Dili	9 : 00		East Timor Time	
Asia/Jayapura	9 : 00		East Indonesia Time	QP0900WIT
Asia/Pyongyang	9 : 00		Korea Standard Time	
Asia/Seoul	9 : 00		Korea Standard Time	QP0900KST
Asia/Tokyo	9 : 00		Japan Standard Time	QP0900UTCS
Asia/Yakutsk	9 : 00	60	Yakutsk Time	
Etc/GMT-9	9 : 00		GMT+09:00	
JST	9 : 00		Japan Standard Time	QP0900JST
Japan	9 : 00		Japan Standard Time	
Pacific/Palau	9 : 00		Palau Time	
ROK	9 : 00		Korea Standard Time	
ACT	9 : 30		Central Standard Time (Northern Territory)	
Australia/Adelaide	9 : 30	60	Central Standard Time (South Australia)	QP0930ACST
Australia/Broken_Hill	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
Australia/Darwin	9 : 30		Central Standard Time (Northern Territory)	
Australia/North	9 : 30		Central Standard Time (Northern Territory)	
Australia/South	9 : 30	60	Central Standard Time (South Australia)	
Australia/Yancowinna	9 : 30	60	Central Standard Time (South Australia/New South Wales)	
AET	10 : 00	60	Eastern Standard Time (New South Wales)	QP1000AEST
Antarctica/DumontDUrville	10 : 00		Dumont-d'Urville Time	
Asia/Sakhalin	10 : 00	60	Sakhalin Time	
Asia/Vladivostok	10 : 00	60	Vladivostok Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Australia/ACT	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Brisbane	10 : 00		Eastern Standard Time (Queensland)	
Australia/Canberra	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Hobart	10 : 00	60	Eastern Standard Time (Tasmania)	
Australia/Lindeman	10 : 00		Eastern Standard Time (Queensland)	
Australia/Melbourne	10 : 00	60	Eastern Standard Time (Victoria)	
Australia/NSW	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Queensland	10 : 00		Eastern Standard Time (Queensland)	
Australia/Sydney	10 : 00	60	Eastern Standard Time (New South Wales)	
Australia/Tasmania	10 : 00	60	Eastern Standard Time (Tasmania)	
Australia/Victoria	10 : 00	60	Eastern Standard Time (Victoria)	
Etc/GMT-10	10 : 00		GMT+10:00	
Pacific/Guam	10 : 00		Chamorro Standard Time	QP1000UTCS
Pacific/Port_Moresby	10 : 00		Papua New Guinea Time	
Pacific/Saipan	10 : 00		Chamorro Standard Time	
Pacific/Truk	10 : 00		Truk Time	
Pacific/Yap	10 : 00		Yap Time	
Australia/LHI	10 : 30	30	Load Howe Standard Time	
Australia/Lord_Howe	10 : 30	30	Load Howe Standard Time	
Asia/Magadan	11 : 00	60	Magadan Time	
Etc/GMT-11	11 : 00		GMT+11:00	
Pacific/Efate	11 : 00		Vanuatu Time	
Pacific/Guadalcanal	11 : 00		Solomon Is. Time	QP1100UTCS
Pacific/Kosrae	11 : 00		Kosrae Time	
Pacific/Noumea	11 : 00		New Caledonia Time	
Pacific/Ponape	11 : 00		Ponape Time	
SST	11 : 00		Solomon Is. Time	
Pacific/Norfolk	11 : 30		Norfolk Time	
Antarctica/McMurdo	12 : 00	60	New Zealand Standard Time	

Time zone ID	Raw offset (Hours : Minutes)	DST offset (Minutes)	Display name	QTIMZON variable (i5/OS only)
Antarctica/South_Pole	12 : 00	60	New Zealand Standard Time	
Asia/Anadyr	12 : 00	60	Anadyr Time	
Asia/Kamchatka	12 : 00	60	Petropavlovsk-Kamchatski Time	
Etc/GMT-12	12 : 00		GMT+12:00	
Kwajalein	12 : 00		Marshall Islands Time	
NST	12 : 00	60	New Zealand Standard Time	QP1200NZST
NZ	12 : 00	60	New Zealand Standard Time	
Pacific/Auckland	12 : 00	60	New Zealand Standard Time	
Pacific/Fiji	12 : 00		Fiji Time	QN1200UTCS, QP1200UTCS
Pacific/Funafuti	12 : 00		Tuvalu Time	
Pacific/Kwajalein	12 : 00		Marshall Islands Time	
Pacific/Majuro	12 : 00		Marshall Islands Time	
Pacific/Nauru	12 : 00		Nauru Time	
Pacific/Tarawa	12 : 00		Gilbert Is. Time	
Pacific/Wake	12 : 00		Wake Time	
Pacific/Wallis	12 : 00		Wallis & Futuna Time	
NZ-CHAT	12 : 45	60	Chatham Standard Time	
Pacific/Chatham	12 : 45	60	Chatham Standard Time	QP1245UTCS
Etc/GMT-13	13 : 00		GMT+13:00	
Pacific/Enderbury	13 : 00		Phoenix Is. Time	
Pacific/Tongatapu	13 : 00		Tonga Time	
Etc/GMT-14	14 : 00		GMT+14:00	
Pacific/Kiritimati	14 : 00		Line Is. Time	

Web module or application server stops processing requests

If an application server process spontaneously closes, or Web modules stop responding to new requests, it is important that you quickly determine why this stoppage is occurring. You can use some of the following techniques to determine whether the problem is a Web module problem or an application server environment problem.

If an application server process spontaneously closes, or Web modules running on the application server stop responding to new requests:

- Try to isolate the problem by installing the Web modules on different servers, if possible.
- If an application server's process spontaneously closes, there will be an SDUMP for you to analyze.
- Use the Tivoli® performance viewer to determine if any of the application server resources, such as the Java heap, or database connections, have reached their maximum capacity. If there is a resource problem, review the application code for a possible cause:

- If database connections are being assigned to a request but are not being released when the requests finish processing, ensure that the application code performs a **close()** on any opened **Connection** object within a **finally{}** block.
- If there is a steady increase in servlet engine threads in use, review application **synchronized** code blocks for possible deadlock conditions.
- If there is a steady increase in a JVM heap size, review application code for memory leak opportunities, such as static (class-level) collections, that can cause objects to never get garbage-collected.
- Enable verbose garbage collection on the application server to help you determine if you have a memory leak problems. This feature adds detailed statements about the amount of available and in-use memory to the JVM error log file.

To enable up verbose garbage collection:

1. In the administrative console, click **Servers > Server Types > Application Servers > *server_name***. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**, and select **Verbose garbage collection**.
2. Stop and restart the application server.
3. Periodically, browse the log file for garbage collection statements. Look for statements beginning with "allocation failure". This string indicates that a need for memory allocation has triggered a JVM garbage collection, to release unused memory. Allocation failures are normal and do not necessarily indicate a problem. However, the statements that follow the allocation failure statement show how many bytes are needed and how many are allocated. If these bytes needed statements indicate that the JVM keeps allocating more memory for its own use, or that the JVM is unable to allocate as much memory as it needs, there might be a memory leak.

IBM Support has documents and tools that can save you time gathering information needed to resolve problems as described in Troubleshooting help from IBM. Before opening a problem report, see the Support page:

- http://www.ibm.com/software/webservers/appserv/zos_os390/support/

Setting a time limit for the completion of RMI/IIOP enterprise bean requests

The Request timeout ORB Service setting determines how long a client waits for a response from an outbound RMI/IIOP enterprise bean invocation. This setting is a server-wide setting that applies to each IIOP locate and request message that is sent as a result of an enterprise bean invocation. When the specified time limit expires, the application that invoked the outbound RMI/IIOP enterprise bean receives an `org.omg.CORBA.COMM_FAILURE` system exception.

Before you begin

Before you begin, you should:

- Determine your settings for all of the dispatch timers. There are separate dispatch timers for message-driven beans (MDBs), (`control_region_mdb_request_timeout`), HTTP requests (`protocol_http_timeout_output`), HTTPS requests (`protocol_https_timeout_output`), SIP requests (`protocol_sip_timeout_output`), SIPS requests (`protocol_sips_timeout_output`), and IIOP requests (`control_region_wlm_dispatch_timeout`). Because enterprise bean invocations can occur while an application is running under an MDB, a servlet or another enterprise bean, you must make sure that the interval setting for the RMI/IIOP outbound timer is shorter than the interval setting for any of these dispatch timers.
- Understand that, from the perspective of the invoker, remote enterprise bean invocations are synchronous. Therefore, while the invoker waits for a response from the enterprise bean, the invoking thread is blocked. When the RMI/IIOP timer expires, the invoking thread is interrupted and returns to the invoker with a system exception response. If a response from the invoked EJB arrives AFTER the RMI/IIOP timer expires, it is ignored.

- Understand the relationship between the RMI/IIOP outbound timer and transactions. When the RMI/IIOP outbound timer expires and a system exception is returned to the invoker, the EJB container immediately puts any existing global transaction into rollback-only state. However, the invoker still returns any response from the target enterprise bean even though the target enterprise bean's transaction is marked for rollback.

About this task

If the timer expires, a message similar to the following message is issued:

```
BB000325W An IIOP request for Class Name 'com.ejb.test.hello.second.EJSRemoteStatelessSayHelloSecond_686a0ff2'
and Method Name 'sayHelloTwo', to 'jobname=BBOS002 asid=0031', has timed out.
SessionHandle=0000000026D9F0480000000A008004FF, Request ID=00000004
```

This message indicates the class and method of the target enterprise bean. If the target enterprise bean was invoked over TCP/IP, the "to" section of the message contains the hostname and the port of the target server. If the target enterprise bean is invoked through optimized local communication, the "to" section of the message contains the target jobname and asid, as shown in the preceding example.

When this message is issued, a corresponding exception trace is generated that includes an entry similar to the following entry:

```
/bbooejsb.cpp+3395 ... BB000011W The function ORBEJSBridge::invoke_request(JNIEnv *, bboojorb *,
char *, CORBA::Boolean, CORBA::Request *&, void *)+3395 received CORBA system exception CORBA::COMM_FAILURE.
Error code is C9C26A48
```

The minor code C9C26A48 in this trace entry indicates that the wait time for the RMI/IIOP outbound timer has ended.

If a response to the request is received after the request times out and is no longer on the queue, the following message is issued:

```
BB000328I: No Request found for inbound GIOP Response,
          SessionHandle=<hstring>, RequestID=<hstring>.
```

A request or reply is uniquely identified by the session handle and the request ID, and can be used to determine if an earlier BB000325W message was received for this request.

To change the length of time that the client waits for a response from an invoked enterprise bean:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name > Container service > ORB Service**.
2. Specify, in seconds, an appropriate timer setting in the Request timeout field. The default value is 180 seconds. Example: Specifying a value of 2 in the Request timeout field sets the wait time for the timer to 2 seconds.

It is recommended that you specify a value that is significantly shorter than the time specified for the dispatch timers. This type of setting enables the invoking application to compensate for nonresponsive enterprise beans before the wait time for the dispatch timers ends. If the wait time for the dispatch timers ends, an EC3 ABEND might occur.

What to do next

Because the org.omg.CORBA.COMM_FAILURE exception is a system exception, the application invoking the enterprise bean is not required to compensate for or retry an expired enterprise bean invocation. However, in certain situations, such as when atomic transactions are not in use by the application, you might want the application to compensate for or retry an expired enterprise bean invocation.

For an application to compensate for or retry an expired enterprise bean invocation, the application must:

- Be running outside of the current global transaction, and

- Catch the `org.omg.CORBA.COMM_FAILURE` exception.

The following example, which is a snippet of code that is running outside an atomic transaction, illustrates the steps an application must perform if you want that application to compensate for or retry an expired enterprise bean invocation:

```
// This method runs outside a global transaction.
public Data callingMethod() throws ... {
    try{
        InitialContext con = new InitialContext();
        EJBHome home = con.lookup(...);
        CalledBean cb = home.create();

    } catch (org.omg.CORBA.COMM_FAILURE cf1){
        // The home create could timeout, so put retry or
        // compensation logic here.

    } catch( CreateException cx){
        throw new ...
    } catch( NamingException nx){
        throw new ...
    } catch(RemoteException ex){
        throw new ...
    }
    try{
        cb.calledMethod(...);

    } catch (org.omg.CORBA.COMM_FAILURE cf2){
// The calledMethod could timeout, so put retry or
// compensation logic here.
    } catch( ... ){
        ...
    }
}

// This method can run in a global transaction.
private void calledMethod(String strKey) throws ... {
    try{
        // business logic here
    }
    catch ( ... ){
        throw new ...
    }
}
```

Applications that run within the scope of an atomic transaction must suspend that transaction before invoking an enterprise bean if you want that application to be able to compensate, for or retry an expired enterprise bean invocation. Embedding the invocation in a `TX_NOTSUPPORTED` enterprise bean method is the best way of suspending the current transaction.

Creating generic servers

A generic server is a server that is managed in the WebSphere Application Server administrative domain even though the server is not a server that is supplied by WebSphere Application Server. The WebSphere Application Server generic servers function enables you to define a generic server as an application server instance within the WebSphere Application Server administration, and associate it with a non-WebSphere WebSphere Application server or process.

About this task

Generic application servers must be non-Java application processes that are either a started task or a shell script. You cannot create a Java application as a generic server for the product.

The following processes can be created as a generic server provided that they are either started tasks or a shell scripts:

- A C or C++ server or process
- A CORBA server
- A Remote Method Invocation (RMI) server

You can use the wsadmin tool or the administrative console to create a generic server.

- **Create a non-Java application as a generic server.** The following steps describe how to use the administrative console to create a non-Java application as a generic application server.

1. Select **Servers > Generic servers**
2. Click **New**.
3. Type in a name for the generic server.

The name must be unique within the node. It is recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Server servers.

4. Click **Next**
5. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.
6. On the **Generic servers** page, click on the name of the generic server.
7. Under Additional Properties, click **Process Definition**.
8. In the Executable name field, enter the name of the non-java process that is launched when you start this generic server.

For example, if you are using a perl script as a generic server, enter the path to the perl.exe module in the Executable name field.

If you have additional arguments, such as the name of the perl script and its parameters, enter them in the Executable arguments field. Multiple arguments must be separated by carriage returns. Use the Enter key on your keyboard to create these carriage returns in the Executable arguments field. The following example illustrates how a perl script application that requires two arguments should appear in this field:

```
perl_application.pl  
arg1  
arg2
```

Note: The Executable target type and Executable target properties are not used for non-Java applications. Executable target type and Executable target properties are only used for Java applications.

9. Click **OK**.

- **Create a Java application as a generic server:** The following steps describe how to use the administrative console to create a Java application as a generic application server.

1. Select **Servers > Server Types > Generic servers**
2. Click **New**.
3. Type in a name for the generic server.

The name must be unique within the node. It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular WebSphere Application Server servers.

4. Click **Next**
5. Click **Finish**. The generic server now appears as an option on the **Application servers** page in the administrative console.
6. Click **Finish**. The generic server now appears as an option on the **Generic servers** page in the administrative console.
7. On the Generic servers page, click on the name of the generic server.

8. Under Additional Properties, click **Process definition**.
9. In the Executable name field under General Properties, enter the path for the WebSphere Application Server default JVM, `{JAVA_HOME}/bin/java`, which is used to run the Java application when you start this generic server.
10. In the Executable target type field under General Properties, select whether a Java class name, **JAVA_CLASS**, or the name of an executable JAR file, **EXECUTABLE_JAR**, is used as the executable target of this Java process. The default value for the product is **JAVA_CLASS**.
11. In the Executable target field under General Properties, enter the name of the executable target. Depending on the executable target type, this is either a Java class containing a `main()` method, or the name of an executable JAR file.) The default value for WebSphere Application Server is `com.ibm.ws.runtime.WsServer`.
12. Click **OK**.

Note: If the generic server is to run on an application server other than a WebSphere Application Server server, leave the Executable name field set to the default value and specify the Java class containing the main function for your application serve in the Executable target field.

What to do next

After you define a generic server, use the Application Server administrative console to start, stop, and monitor the associated non-WebSphere Application Server server or process when stopping or starting the applications that rely on them.

Note: You can use either the **Terminate** or **Stop** buttons in the administrative console to stop any application server, including a generic application server.

Starting and terminating generic application servers

This topic describes how to start and terminate generic servers.

About this task

If you create a generic server on a base WebSphere Application Server, you cannot use the base Application Server administrative console to start or terminate this server. You must use the `wsadmin` tool to manage this server.

If you create a generic server in a Network Deployment environment, you can use the administrative console to start and terminate this server.

1. Start a generic application server.

There are two ways to start a generic server in a Network Deployment environment. You can use the managed bean (MBean) `NodeAgent launchProcess` operation of the `wsadmin` tool, or you can use the administrative console. To use the administrative console:

- a. In the administrative console, click **Servers > Server Types > Generic servers**.
- b. Select the name of the generic server you want to start, and then click **Start**.
- a. View the **Status** value and any messages or logs to see whether the generic server starts.

2. Terminate generic servers.

There are two ways to terminate a generic server in a Network Deployment environment. You can use the MBean `terminate launchProcess` operation of the `wsadmin` tool, or you can use the administrative console. To use the administrative console:

- a. In the administrative console, click **Servers > Server Types > Generic servers**.
- b. Select the check box beside the name of the generic server, and then click **Terminate** or **Stop**.
- c. View the **Status** value and any messages or logs to see whether the generic server terminates.

Generic server settings

Use this page to view or change the settings of a generic server.

A generic server is a server that is managed in the product administrative domain, although it is not a server that is provided with the product. The generic server can be any server or process that is necessary to support the Application Server environment, including a Java server, a C or C++ server or process, or a Remote Method Invocation (RMI) server.

To view this administrative console page, click **Servers > Server Types > Generic servers > *server_name***.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information. The **Runtime** tab is available only when the server is running.

Name

Specifies a logical name for the generic server.

Generic server names must be unique within a node. For multiple nodes within a cluster, you can have different generic servers with the same server name as long as the server and node pair are unique. For example, a server named `server1` in a node named `node1` in the same cluster with a server named `server1` in a node named `node2` is allowed. Configuring two servers named `server1` in the same node is not allowed. The product uses the server name for administrative actions, such as referencing the server in scripting.

It is highly recommended that you use a naming scheme that makes it easy to distinguish your generic application servers from regular product application servers. This will enable you to quickly determine whether to use the Terminate or Stop button in the administrative console to stop a specific application server.

You must use the Terminate button to stop a generic application server.

Data type	String
Default	

Setting up the Server Runtime on multiple systems in a sysplex

If your applications require workload balance, and high availability, and you want the ability to easily add new systems to meet demand as your workload grows, you might want to migrate your Application Server runtime and associated application servers from a monoplex to a sysplex configuration.

Before you begin

After you install the server runtime and associated business application servers on a monoplex, you should evaluate whether you want to migrate the server runtime and associated application servers to a sysplex configuration. If you decide to set up a sysplex environment that consists of several z/OS images and workload management (WLM), you must run the BBOWWPFA job in the z/OS image on which you are starting WebSphere Application Server.

The BBOWWPFA job is one of the jobs that are automatically generated when you initially configure and customization WebSphere Application Server. You run this batch of jobs to set up your z/OS environment. Typically these JCL jobs are executed one by one in the first z/OS image that is available on the sysplex. However, the BBOWWPFA job, which sets up the runtime (configuration) file system for the product must run in the same z/OS image as where you plan to start the product. To ensure that the BBOWWPFA job runs in the proper image, add the following JCL statement, after the BBOWWPFA job statement, within the batch of automatically generated JCL statements, before you run the BBOWWPFA job.

/*JOBPARM SYSAFF=sxx

where sxx is the name of z/OS image in which you are going to run the product.

About this task

A sysplex environment enables you to:

- Balance the workload across multiple systems, thus providing better performance management for your applications.
- Add new systems to meet demand as your workload grows. This capability provides a scalable solution to your processing needs.
- Replicate the runtime and associated business application servers. This capability ensures that if a failure occurs on one system, other systems are available to handle user requests.
- Upgrade the Application Server from one release or service level to another without interrupting service to your users.

Perform the following tasks to setup the Application Server in a sysplex configuration.

1. Set up a sysplex environment if one is not already available.

The z/OS publication *z/OS MVS Setting Up a Sysplex* describes how to set up a z/OS sysplex. The directory you set up should be similar to the following directory structure:

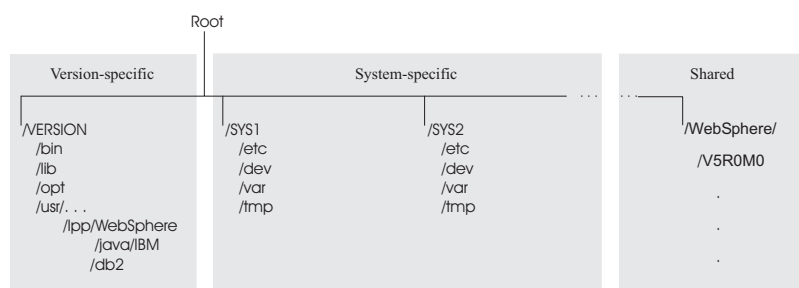


Figure 1. Directory structure for two Application Servers running in a Sysplex

2. Configure the Server runtime for a sysplex environment.

- a. Decide whether you want a single-system view of the error log. If you want a single-system view of the error log, and initially you set up the error log in the system logger and used DASD for logging, you must now configure the error log in the coupling facility.
- b. Decide how you will share application executables in the cell.
- c. Set up ARM. This release does not support cross-system restart, so you must set up your ARM policy accordingly. Make sure you specify TARGET_SYSTEM for the system on which each element runs (if you take the default TARGET_SYSTEM=*, you get cross-system restart).
- d. Decide whether you will run all of the run-time servers on every system in the cell.

Recommendations: The following table provides recommendations and requirements for running servers in a cell.

Table 5. Running servers in a cell

Server	Recommendations and requirements for running servers in a cell
location service daemon and node agent	<ul style="list-style-type: none"> • You must run both the location service daemon and the node agent on each system in the sysplex in which you want the server runtime to run. If the server runtime is not installed on some of the systems in your sysplex, you do not have to run a location service daemon and node agent on those systems. • If a server indicates that PassTickets are desirable for interaction with a client, you must run the location service daemon and node agent on the system where the z/OS client resides.

Table 5. Running servers in a cell (continued)

Server	Recommendations and requirements for running servers in a cell
deployment manager	Make sure you follow the correct steps to configure a deployment manager cell.

3. Prepare your security system.
4. Set up data sharing. Refer to the *DB2 Data Sharing: Planning and Administration* publication for the version of DB2 that is running on your z/OS system.
5. Customize z/OS functions on the other systems in the sysplex to match the customization you performed as part of the initial server runtime installation.
6. Change the TCP/IP settings. Each system in a sysplex contains a location service daemon, node agent, and business application servers. The location service daemon acts as a location service agent and accepts locate requests with object keys in the requests. Therefore it is important that the TCP/IP domain name server (DNS) entries, and TCP/IP profile for each system in the cell include the port for the location service daemon, and that port is associated with the name of the new location service daemon server.

- a. Change DNS entries.

If you use a DNS implementation that allows the use of generic IP names that dynamically resolve to like-configured servers, you must adjust the IP names in your DNS. You must keep the generic IP name of the location service daemon, but add a new IP name for the second and subsequent location service daemon servers. The additional IP names enable the DNS to direct work to other servers if a failure occurs.

- b. In the TCP/IP profile for each additional system in the cell, add a port for the location service daemon, and associate that port with a new location service daemon server name.

By default, the server runtime uses port 5655 for the location service daemon. The server runtime also names the first location service daemon server DAEMON01 and increments the suffix on that name for each new location service daemon server; for example, DAEMON02, DAEMON03, and so forth. Therefore, for the second system in the sysplex, you must add a port and associate it with DAEMON02.

Example:

```
5655
TCP      DAEMON02
```

Follow the same pattern for the third and subsequent systems in the sysplex.

7. Define new application server clusters in the sysplex.
8. Optional: Create deployment manager cells.
 - a. Install the default application server on each node in your sysplex.
 - b. Install a deployment manager cell on one node in your sysplex.
 - c. Add default server nodes to the deployment manager cell.

Results

You can take advantage of all of the benefits of running your applications on multiple systems in a sysplex.

What to do next

Migrate your applications to the sysplex.

Customizing base z/OS functions on the other systems in a sysplex

Some z/OS system functions, like the Language Environment (LE) and DB2, must be customized to properly interface with the server runtime. During the server runtime installation process, you use either the Profile Management Tool or the `zpm` command to customize these functions on the system where the

server runtime is initially installed. If you want to use the server runtime in a sysplex environment, you must make sure that the same customization changes are made to all of the other systems in the sysplex, on which the server runtime is installed.

Before you begin

Create copies of the sample files.

About this task

The following steps help you customize the z/OS system functions on the other systems in the sysplex. The resulting customization should mirror the customization on the initial system.

The following steps assume that your default server runtime data set high-level qualifier (hlq) is BB0. If BB0 is not your default server runtime data set hlq, modify the following steps to use your specified hlq.

1. If you are running on z/OS Version 1.8.x or lower, update the SCHEDxx file to include the statements from the BBOSCHED sample file. You do not have to perform this step if you are running on z/OS Version 1.9 or higher because, starting with Version 1.9, a PPT entry already exists for BPXBATA2.
2. Ensure that the Language Environment (LE) data sets SCEERUN and SCEERUN2, and the DB2 data set SDSNLOAD are APF-authorized.
3. Place the server runtime modules in the appropriate link pack area (LPA) or link list.

The following table indicates where to place each module.

Table 6. Placing modules in LPA or link list

Modules	Notes
WAS_HOME/lib/ipcs/ BBO.SBBOULIB	You can put members into the link list or LPA. Do not place these members in either the LPA or link list.
Rule: These data sets are PDSEs and cannot be added to members in LPALSTxx or IEALPAxx.	
Recommendation: For automation, if you want to ensure that the server runtime modules are loaded into dynamic LPA and are available after an IPL, create a new PROGxx member with the SETPROG LPA commands and invoke the PROGxx member from the PARMLIB COMMNDxx.	
Example:	
<pre> SETPROG LPA,ADD,MASK=*,DSNAME= BB0. SBBOLOAD SETPROG LPA,ADD,MASK=*,DSNAME= BB0. SBBOLPA </pre> <ul style="list-style-type: none"> • Change BB0 to the appropriate qualifier if BB0 is not the high-level qualifier for your server runtime data sets. • If you are using SETPROG on a running system, before you add a module, you must purge any already existing module, that has the same name, from the LPA. 	

4. Optional: If you used a PROGxx file for APF authorizations or the LPA, must issue the following command:
SET PROG=xx
5. Optional: Make sure that all of the BBO.* data sets are cataloged.
6. Update your SYS1.PARMLIB(BLSCUSER) member with the IPCS models supplied by member BBOIPCSP. For more information about the SYS1.PARMLIB(BLSCUSER) member, see *z/OS MVS IPCS User's Guide*.
7. Optional: Start System Management Facility (SMF) recording. If you want to start SMF recording to collect system and job-related information:

- a. Edit the SMFPRMxx parmlib member.
 - 1) Insert an 'ACTIVE' statement to indicate SMF recording.
 - 2) Insert a SYS statement to indicate the types of SMF records that you want the system to create.

Example: Use SYS(TYPE(120:120)) to select type 120 records only. Keep the number of selected record types small, to minimize the performance impact.
- b. Issue the following command to start writing records to a Direct Access Storage Device (DASD):


```
t smf=xx
```

Where xx is the suffix of the SMF parmlib member (SMFPRMxx). For more information about the SMF parmlib member, see *z/OS MVS System Management Facilities (SMF)*.

When you activate writing to DASD, the data is recorded in a data set (specified in SMFPRMxx).

Results

The same z/OS system function customization exists on all of the systems in the sysplex on which the server runtime is installed.

Configuring transport chains

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Before you begin

Ensure that a port is available for the new transport chain. If you need to set up a shared port, you must:

- Use wsadmin commands to create your transport chain.
- Make sure that all channels sharing that port have the same discrimination weight assigned to them.

About this task

You need to configure transport chains to provide networking services to such functions as the service integration bus component of IBM service integration technologies, the caching proxy, and the high availability manager core group bridge service.

You can use either the administrative console or wsadmin commands to create a transport chain. If you use the administrative console, complete the following steps:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name*** or **Servers > Server Types > WebSphere proxy servers > *server_name***, and then select one of the following options, depending on the type of chain you are creating:

For application servers, in the Container settings section select one of the following options:

- Click **SIP Container Settings > SIP container transport chains**.
- Click **Web container settings > Web container transport chains**.
- Click **Container services > ORB service > ORB service transport chains**.
- In the Server messaging section, click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.

For proxy servers, under HTTP proxy server settings, click **Proxy server transports** and select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**. Then click **HTTP proxy inbound channel**.

2. Click **New**.

The Create New Transport Chain wizard initializes. During the transport chain creation process, you are asked to:

- Specify a name for the new chain.
- Select a transport chain template
- Select a port, if one is available to which the new transport chain is bound. If a port is not available or you want to define a new port, specify a port name, the host name or IP address for that port, and a valid port number.

Note: If you are configuring a chain that contains a TCP channel, the wizard displays a list of configured TCP channels and a list of the ports that the listed TCP channels are not using. You must select one of the ports that none of the other TCP channels are using.

Similarly, if you are configuring a transport chain that contains a UDP channel, the wizard displays a list of already configured UDP channels and a list of the ports that these UDP channels are not using. You must select one of the ports that none of the other UDP channels are using.

When you click **Finish**, the new transport chain is added to the list of defined transport chains on the **Transport chain** panel.

3. Click the name of a transport chain to view the configuration settings that are in effect for the transport channels contained in that chain.

To change any of these settings, complete the following actions:

- a. Click the name of the channel whose settings you need to change.
- b. Change the configuration settings.

Some of the settings, such as the port number, are determined by what is specified for the transport chain when it is created and cannot be changed.

- c. Click on **Custom properties** to set any custom properties that are defined for your system.

4. When you your configuration changes, click **OK**.

5. Stop the application server and start it again.

You must stop the application server and start it again before your changes take effect.

What to do next

Update any routines you have that issue a call to start transports during server startup. When a routine issues a call to start transports during server startup, the product issues an error message.

Transport chains

Transport chains represent a network protocol stack that is used for I/O operations within an application server environment.

Transport chains are part of the channel framework function that provides a common networking service for all components, including the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service.

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, DCS, or HTTP. Network ports can be shared among all of the channels within a chain. The channel framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

Note: If you have a routine that issues a call to start transports during server startup, unless you have a mixed-node environment and that server is running in either a Version 5.1 or a Version 6.x node,

you must modify your routine to issue a call to start transport chains instead of the transports. The product issues an error message if it receives a call to start transports for a server that is not running in a Version 5.1 or a Version 6.x node.

The transport chain configuration settings determine which I/O protocols are supported for that chain. Following are some of the more common types of channels. Custom channels that support requirements unique to a particular customer or environment can also be added to a transport chain.

DCS channel

Used in a Network Deployment environment by the core group bridge service, the data replication service (DRS), and the high availability manager to transfer data, objects, or events among application servers.

HTTP inbound channel

Used to enable communication with remote servers. It implements the HTTP 1.0 and 1.1 standards and is used by other channels, such as the Web container channel, to serve HTTP requests and to send HTTP specific information to servlets expecting this type of information.

HTTP inbound channels are used instead of HTTP transports to establish the request queue between a Web server plug-in, and a Web container in which the Web modules of an application reside.

HTTP proxy inbound channel

Used to handle HTTP requests between a proxy server and application server nodes.

HTTP Tunnel channel

Used to provide client applications with persistent HTTP connections to remote hosts that are either blocked by firewalls or require an HTTP proxy server, including authentication, or both. An HTTP Tunnel channel enables the exchange of application data in the body of an HTTP request or response that is sent to or received from a remote server. An HTTP Tunnel channel also enables client-side applications to poll the remote host and to use HTTP requests to either send data from the client or to receive data from an application server. In either case, neither the client nor the application server is aware that HTTP is being used to exchange the data.

JFAP channel

Used by the Java Message Service (JMS) server to create connections to JMS resources on a service integration bus.

MQ channel

Used in combination with other channels, such as a TCP channel, within the confines of WebSphere MQ support to facilitate communications between a WebSphere System Integration Bus and a WebSphere MQ client or queue manager.

ORB Service channel

Used in combination with other channels, such as a TCP channel, to handle CORBA and RMI/IIOP messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

SIP channel

Used to create a bridge in the transport chain between a session initiation protocol (SIP) inbound channel, and a servlet and JavaServer Page engine.

SIP container inbound channel

Used to handle communication between the SIP inbound channel and the SIP servlet container.

SIP inbound channel

Used to handle inbound SIP requests from a remote client.

SSL channel

Used to associate an Secure Sockets Layer (SSL) configuration repertoire with the transport chain. This channel is only available when SSL support is enabled for the transport chain. An SSL

configuration repertoire is defined in the administrative console, under security, on the **SSL configuration repertoires > SSL configuration repertoires** page.

TCP channel

Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses transmission control protocol (TCP) to retrieve information from a network.

UDP channel

Used to provide client applications with persistent connections within a Local Area Network (LAN) when a node uses user datagram protocol (UDP) to retrieve information from a network.

Web container channel

Used to create a bridge in the transport chain between an HTTP inbound channel and a servlet and JavaServer Page (JSP) engine.

HTTP transport collection

Use this page to view or manage HTTP transports. Transports provide request queues between Web server plug-ins and Web containers in which the Web modules of applications reside. When you request an application in a Web browser, the request is passed to the Web server, then along the transport to the Web container.

Note: You can use HTTP transports only on a Version 5.1 or V6.0.x node in a mixed cell environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

To view the HTTP Transport administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name* > Web container settings > Web container > HTTP transports**.

Host

Specifies the host IP address to bind for transport. If the application server is on a local machine, the host name might be localhost.

Port

Specifies the port to bind for transport. The port number can be any port that currently is not in use on the system. The port number must be unique for each application server instance on a given machine.

For z/OS, a maximum of two ports, one for HTTP requests and one for HTTPS requests, is allowed for each process configured as an HTTP transport. Additional ports can be configured as HTTP transport chains. Additional ports can be configured as HTTP transport channels.

SSL Enabled

Specifies whether to protect transport connections with Secure Sockets Layer (SSL). The default is not to use SSL.

HTTP transport settings

Use this page to view and configure an HTTP transport. The name of the page might be that of an SSL setting such as DefaultSSLSettings. This page is not available if you do not have an HTTP transport defined for your system.

Note: You can use HTTP transports only on a Version 5.1 or V6.0.x node in a mixed cell environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

If you have HTTP transports defined for your system, in the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***, and then in the Container Settings section, click **Web container > HTTP transports > *host_name*** to view or change the settings for your HTTP transport.

Host

Specifies the host IP address to bind for transport.

If the application server is on a local machine, the host name might be localhost.

Data type	String
------------------	--------

Port

Specifies the port to bind for transport. Specify a port number between 1 and 65535. The port number must be unique for each application server on a given machine.

Data type	Integer
Range	1 to 65535

SSL Enabled

Specifies whether to protect transport connections with Secure Sockets Layer (SSL). The default is not to use SSL.

Data type	Boolean
Default	false

SSL

Specifies the Secure Sockets Layer (SSL) settings type for SSL connections. The options include one or more SSL settings that are defined in the Security Center; for example, DefaultSSLSettings, ORBSSLSettings, or LDAPSSLSettings.

Data type	String
Default	An SSL setting defined in the Security Center

HTTP transport custom properties

You can use the administrative console to set custom properties for an HTTP transport. The HTTP transport custom properties administrative console page only appears if you have an HTTP transport defined for your system.

Note: You can use HTTP transports only on a V5.1 node in a mixed cell environment. You must use HTTP transport channels instead of HTTP transports to handle your HTTP requests on all of your other nodes. The topic *HTTP Tunnel transport channel custom property* describes the custom properties that you can specify for an HTTP transport channel.

The use of IPv6 (Internet Protocol Version 6) and WS-AT (Web Services Atomic Transactions) are not supported on HTTP transports; they are only supported on HTTP transport channel chains.

If you are using HTTP transports, you can set the following custom properties on either the Web container or HTTP transport custom properties page in the administrative console. When set on the Web container custom properties page, all transports inherit the properties. Setting the same properties on a transport overrides like settings defined for a Web container.

To specify custom properties for a specific transport on the HTTP transport:

1. In the administrative console click **Servers > Server Types > WebSphere application servers > server_name**.
2. Then in the Container Settings section, click **Web container > Web container settings > HTTP transport**.
3. Select a host.
4. In the Additional Properties section, select **Custom Properties**.
5. On the custom properties page, click **New**.
6. On the settings page, enter the property you want to configure in the **Name** field and the value you want to set it to in the **Value** field.
7. Click **Apply** or **OK**.
8. Click **Save** on the console task bar to save your configuration changes.
9. Restart the server.

Following is a list of custom properties provided with the product. These properties are not shown on the settings page for an HTTP transport.

ConnectionIOTimeout:

Use the ConnectionIOTimeout property to specify how long the J2EE server waits for an I/O operation to complete. Set this variable for each of the HTTP transport definitions on the server. You will need to set this variable for both SSL transport and non-SSL transport. Specifying a value of zero disables the time out function.

Data type	Integer
Default	120 seconds for the z/OS platform

ConnectionKeepAliveTimeout:

Use the ConnectionKeepAliveTimeout property to specify the maximum number of seconds to wait for the next request on a keep alive connection.

Data type	Integer
Default	30 seconds for the z/OS platform

ConnectionResponseTimeout:

Use the ConnectionResponseTimeout property to set the maximum amount of time, in seconds, that the server waits for an application component to respond to an HTTP request. Set this variable for each of the HTTP transport definitions on the server. You must set this variable for both SSL transport and non-SSL transport. If the response is not received within the specified length of time, the servant might fail with ABEND EC3 and RSN=04130007. Setting this timer prevents client applications from waiting for a response from an application component that might be deadlocked, looping, or encountering other processing problems that cause the application component to hang.

This property can be set for both HTTP transports and HTTP transport channels.

Use the server custom properties protocol_http_timeout_output_recovery, and protocol_https_timeout_output_recovery, to indicate the recovery action that you want taken on timeouts for requests received over the HTTP and HTTPS transports.

Data type	Integer
Default	300 seconds

MaxKeepAliveRequests:

Use the `MaxKeepAliveRequests` property to specify the maximum number of requests which can be processed on a single keep alive connection. This parameter can help prevent denial of service attacks when a client tries to hold on to a keep-alive connection. The Web server plug-in keeps connections open to the application server as long as it can, providing optimum performance.

On the z/OS platform, when this property is set to 0 (zero), the connection is closed after every request.

Data type	Integer
Default	50 requests for the z/OS platform

MutualAuthCBindCheck:

This property is only valid on the z/OS platform. Use the `MutualAuthCBindCheck` property to specify whether or not a client certificate should be resolved to a SAF principal. If this property is set to `true`, all SSL connections from a browser must have a client certificate, and the user ID associated with that client certificate must have RACF CONTROL authority for `CB.BIND.servername`. If these conditions are not met, the connection will be closed. Issue the following RACF command to give the user ID associated with that client certificate RACF CONTROL authority:

```
PERMIT CB.BIND.servername CLASS(CBIND) ID(clientCertUserId) ACCESS(CONTROL)
```

Data type	String
Value	true or false
Default	false

RemoveServerHeader: Use this property to specify whether an existing server header is removed before a response message is sent. If this property is set to `true`, the value specified for the `ServerHeaderValue` property is ignored.

Data type	String
Value	true or false
Default	false

ResponseBufferSize:

This property is used to specify, in bytes, the default size of the initial buffer allocation for the response buffer. When the buffer fills up, a flush for this buffer space will automatically occur. If a value is not specified for this property, the default response buffer size of 32K bytes is used.

The `setBufferSize()` API method can be used to override the value specified for this custom property at the individual servlet level.

Data type	Integer
Default	32000 bytes

ServerHeader:

This property is only valid for the z/OS platform. Use the `ServerHeader` property to suppress the server HTTP header (`Server:`) in responses. When the server header custom property is not specified, the default is equal to a setting of `true` and the server header is included in the HTTP response. Set this property to `false` if you want to prevent the inclusion of the server header.

Data type	String
Value	true or false
Default	true

ServerHeaderValue: Use this property to specify a server header this is added to outgoing response messages if server header is not already provided. This property is ignored if the RemoveServerHeader property is set to true.

Data type	string
Default	WebSphere Application Server/x.x

x.x is the version of WebSphere Application Server that you are using.

SoLingerValue: Use this property to specify, in seconds, the amount, that the socket close operation waits for data contained in the TCP/IP send buffer to be sent. This property is ignored if the UseSoLinger property is set to false.

Data type	Integer
Default	20 seconds

TcpNoDelay: Use this property to set the socket TCP_NODELAY option which enables and disables the use of the TCP Nagle algorithm for connections received on this transport. When this property is set to true, use of the Nagle algorithm is disabled.

Data type	String
Value	true or false
Default	true

Trusted: Use the Trusted property to indicate that the application server can use the private headers that the Web server plug-in adds to requests.

Data type	String
Value	true or false
Default	false

Note: This property must be set to false for Secure Sockets Layer (SSL) client certificate authentication to work.

UseSoLinger: Use this property to set the socket SO_LINGER option. This property configures whether the socket close operation waits until all of the data contained in the TCP/IP send buffer is sent before closing a connection. If this property is set to true, and the time expires before the all of the content of the send buffer sent, any data remaining in the send buffer is lost.

The SoLingerValue property is ignored if this property is set to false.

Data type	String
Value	true or false
Default	true

Transport chains collection

Use this page to view or manage transport chains. Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP or HTTP. Network ports can be shared among all of the channels within a chain. The Channel Framework function automatically distributes a request arriving on that port to the correct I/O protocol channel for processing.

The **Transport chains** page lists the transport chains defined for the selected application server. Transport chains represent network protocol stacks operating within this application server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name > Ports**. Click on **View associated transports** for the port whose transport chains you want to view.

Name

Specifies a unique identifier for the transport chain. The name must consist of alphanumeric or national language characters and can start with a number. The name must be unique within the product configuration. Click on the name of a transport chain to change its configuration settings.

Enabled

When set to true, indicates that the transport chain is activated at application server startup.

Host

Specifies the host IP address to bind for the transport chain. If the application server is on a local machine, the host name might be localhost.

Port

Specifies the port to bind for the transport chain. The port number can be any port that currently is not in use on the system, might be localhost or the wildcard character * (an asterisk). The port number must be unique for each application server instance on a given machine

SSL Enabled

When enabled, users are notified that there is a channel that enables Secure Sockets Layer (SSL) in the listed transport chain. When SSL is enabled, all traffic going through this transport is encrypted and digitally secured.

Transport chain settings

Use this page to view a list of the types of transport channels configured for the selected transport chain. A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

To view this administrative console page, click **Servers > Server Types** , and then click either **WebSphere application servers** or **WebSphere proxy servers**. Click a server name, and then click **Ports > View associated transports** for the port whose transport chains you want view, and then click the name of a specific chain.

Name

Specifies the name of the selected transport chain.

You can edit this field to rename this transport chain. However, remember that the name must be unique within the product configuration.

Enabled

When checked, this transport chain is activated at application server or proxy server startup.

Transport channels

Lists the transport channels configured for this transport chain and their configuration settings. Click the name of a transport channel to view the configuration settings for that channel.

HTTP tunnel transport channel settings

Use this page to view and configure an HTTP tunnel transport channels. Inbound connections sent through this channel are tunneled over HTTP, allowing intermediates to view this data as the body of an HTTP message instead of in its natural format. This type of channel is often used to circumvent firewalls with protocol restrictions.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name* > Ports**. Click on **View associated transports** for the port associated with the HTTP Tunnel transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the HTTP tunnel transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels within the product environment. For example, an HTTP tunnel transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type string

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type Positive integer

Default 0

HTTP transport channel settings

Use this page to view and configure an HTTP transport channel. This type of transport channel handles HTTP requests from a remote client.

An HTTP transport channel parses HTTP requests and then finds an appropriate application channel to handle the request and send a response.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name* > Ports**. Locate the port for the HTTP channel whose settings you want to view or configure, and click **View associated transports**. Click the name of the transport chain that includes this HTTP transport, and then click the name of the HTTP transport channel.

Transport channel name

Specifies the name of the HTTP transport channel.

The name field cannot contain any of the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in your system. For example, an HTTP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled, and the transport chain includes multiple channels to which it might forward data. The channel in the chain that has the lowest discrimination weight is the first channel that looks at incoming data and determines whether it owns that data.

Data type Positive integer
Default 0

Read timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits for a read request to complete on a socket after the first read occurs. The read being waited for could be part of the body of the read request, such as a POST, or part of the headers, if all of the headers are not read as part of the first read that occurs on the socket for this request.

Note: The value specified for this property, in conjunction with the value specified for the Write timeout property, provides the timeout functionality that the ConnectionIOTimeout custom property provided in previous releases.

Data type Integer
Default 60 seconds

Write timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits on a socket for each portion of the response data to be transmitted. This timeout typically only occurs in situations where the writes are lagging behind new requests. This situation can occur when a client has a low data rate or the network interface card (NIC) for the server is saturated with I/O.

Note: The value specified for this property, in conjunction with the value specified for the Read timeout property, provides the timeout functionality that the ConnectionIOTimeout custom property provided in previous releases.

If some of your clients require more than 300 seconds to receive data being written to them, change the value specified for the Write timeout parameter. Some clients are slow and require more than 300 seconds to receive data that is sent to them. To ensure they are able to obtain all of their data, change the value specified for this parameter to a length of time in seconds that is sufficient for all of the data to be received. Make sure that if you change the value of this setting, that the new value still protects the server from malicious clients.

Data type Integer
Default 300 seconds

Persistent timeout

Specifies the amount of time, in seconds, that the HTTP transport channel allows a socket to remain idle between requests.

Note: The value specified for this property provides the timeout functionality that the ConnectionKeepAliveTimeout custom property provided in previous releases.

Data type Integer
Default 30 seconds

Use persistent (keep-alive) connections

When selected, specifies that the HTTP transport channel connections are left open between requests. Leaving the connections open can save setup and tear down costs of sockets if your workload has clients that send multiple requests.

If your clients only send single requests over substantially long periods of time, it is probably better to disable this option and close the connections right away rather than to have the HTTP transport channel setup the timeouts to close the connection at some later time.

The default value is true, which is typically the optimal setting.

Note: If a value other than 0 is specified for the maximum persistent requests property, the Use persistent (keep-alive) connections property setting is ignored.

Unlimited persistent requests per connection

When selected, specifies that the number of persistent requests per connection is not limited.

Maximum persistent requests per connection

When selected, specifies that the number of persistent requests per connection is limited to the number specified for the Maximum number of persistent requests property. This property setting is ignored if the Use persistent (keep-alive) connections property is not enabled.

Change the value specified for the Maximum persistent requests parameter to increase the number of requests that can flow over a connection before it is closed. When the Use persistent connections option is enabled, the Maximum persistent requests parameter controls the number of requests that can flow over a connection before it is closed. The default value is 100. This value should be set to a value such that most, if not all, clients always have an open connection when they make multiple requests during the same session. A proper setting for this parameter helps to eliminate unnecessary setting up and tearing down of sockets.

For test scenarios in which the client will never close a socket or where sockets are always proxy or Web servers in front of your application server, a value of -1 disables the processing, which limits the number of requests over a single connection. The persistent timeout still shuts down some idle sockets and protect your server from running out of open sockets.

Maximum persistent requests per connection

Specifies the maximum number of persistent requests that are allowed on a single HTTP connection. You can add a value to this field only if the **Maximum persistent requests per connection** property is selected.

When the Use persistent connections option is enabled, the Maximum persistent requests parameter controls the number of requests that can flow over a connection before it is closed. The default value is 100. This value should be set to a value such that most, if not all, clients always have an open connection when they make multiple requests during the same session. A proper setting for this parameter helps to eliminate unnecessary setting up and tearing down of sockets.

For test scenarios in which the client will never close a socket or where sockets are always proxy or Web servers in front of your application server, a value of -1 will disable the processing which limits the number of requests over a single connection. The persistent timeout will still shutdown some idle sockets and protect your server from running out of open sockets.

If a value of 0 or 1 is specified, only one request is allowed per connection.

Data type	Integer
Default	100

Maximum header field size

Specifies, in bytes, the maximum size for a header that can be included on an HTTP request.

Setting this property to a realistic size for your applications helps you to prevent denial of service (DoS) attacks that use large headers within an HTTP request as an attempt to make a system resource, such as the applications that handle HTTP requests, essentially unavailable to intended users.

The default for this property is 32768 bytes.

Maximum headers

Specifies the maximum number of headers that can be included in a single HTTP request.

Setting this property to a realistic number for your applications helps you to prevent denial of service (DoS) attacks that use a large number of headers within an HTTP request as an attempt to make a system resource, such as the applications that process HTTP requests, essentially unavailable to their intended users.

The default for this property is 50.

Limit request body buffer size

When selected, specifies that size of the body of an HTTP request is limited.

This property can be used to prevent denial of service attacks that use large HTTP requests as an attempt to make a system resource, such as the applications that process HTTP requests, essentially unavailable to their intended users.

Maximum request body buffer size

Specifies, in bytes, the maximum size limit for the body of an HTTP request. If this size is exceeded, the request is not processed.

A value can be added to this field only if the **Limit request body buffer size** property is selected.

Logging

You can use the settings in this section to configure and enable National Center for Supercomputing Applications (NCSA) access logging, or HTTP error logging. If you are running the product on z/OS, you can also use this section to configure and enable Fast Response Cache Accelerator (FRCA) logging. Enabling any of these logging services slows server performance.

If you want any of the enabled logging services to start when the server starts, click **Servers > Server Types > WebSphere application servers > server_name**. Then in the Troubleshooting section, click **HTTP error, NCSA access and FRCA logging**, and select **Enable logging service at server start-up**. When this option is selected, any HTTP error, NCSA or FRCA logging service that is enabled automatically starts when the server starts.

Note: If you are running the product on z/OS, HTTP error, NCSA access, and FRCA logging settings must be specified on the controller. These settings are ignored if they are specified on the servant or adjunct.

NCSA access logging

By default, the **Use global logging service** option is selected for NCSA access logging. This setting means that the NCSA access logging settings default to the settings specified for NCSA access logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **NCSA Access logging** section, and select the **Use chain-specific logging** option.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:

- Explicitly enable or disable NCSA access logging.
- Specify an access log file path that is different from the default path.
- Specify a maximum size for the access log file that is different from the default maximum size.
- Explicitly select the format of the NCSA access log file.

Enable access logging

When selected, a record of inbound client requests that the HTTP transport channel handles is kept in the NCSA access log file.

Access log file path

Specifies the directory path and name of the NCSA access log file. Standard variable substitutions, such as `$(SERVER_LOG_ROOT)`, can be used when specifying the directory path.

If you are running the product on z/OS, you should use a server-specific variable, such as `$(SERVER_LOG_ROOT)`, to prevent log file name collisions.

Access log maximum size

Specifies the maximum size, in megabytes, of the NCSA access log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, the file is overwritten with the most current version of the original log file.

Maximum number of historical files

Specifies the maximum number of historical versions of the NCSA access log file that are kept for future reference.

NCSA access log format

Specifies in which format the client access information appears in the NCSA log file. If Common is selected, the log entries contain the requested resource and a few other pieces of information, but does not contain referral, user agent, and cookie information. If Combined is selected, referral, user agent, and cookie information is included.

FRCA logging

By default, the **Use global logging service** option is selected for FRCA logging. This setting means that the FRCA logging settings default to the settings that are specified for NCSA access logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **FRCA logging** section, and select the **Use chain-specific logging** option.

This field only displays if you are running the product on z/OS.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:

- Explicitly enable or disable FRCA logging.
- Specify an access log file path that is different from the default path.
- Specify a maximum size for the access log file that is different from the default maximum size.
- Explicitly select the format of the FRCA log file.

Enable FRCA logging

When selected, a record of inbound client requests that the HTTP transport channel handles is kept in the FRCA log file.

This field only displays if you are running the product on z/OS.

FRCA log file path

Specifies the directory path and name of the FRCA log file. you should use a server-specific variable, such as `$(SERVER_LOG_ROOT)`, to prevent log file name collisions.

This field only displays if you are running the product on z/OS.

FRCA log maximum size

Specifies the maximum size, in megabytes, of the FRCA log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, the file is overwritten with the most current version of the original log file.

This field only displays if you are running the product on z/OS.

Maximum number of historical files

Specifies the maximum number of historical versions of the FRCA log file that are kept for future reference.

This field only displays if you are running the product on z/OS.

FRCA log format

Specifies in which format the client access information appears in the FRCA log file. If Common is selected, the log entries contain the requested resource and a few other pieces of information, but does not contain referral, user agent, and cookie information. If Combined is selected, referral, user agent, and cookie information is included.

This field only displays if you are running the product on z/OS.

Error logging

By default, the **Use global logging service** option is selected for Error logging. This setting means that the Error logging settings default to the settings that are specified for Error logging on the **HTTP error, NCSA access and FRCA logging** page in the administrative console. If you want to change these settings for this specific HTTP transport channel, expand the **Error logging** section, and select the **Use chain-specific logging** option.

After you select the **Use chain-specific logging** option, you can make the following configuration changes:

- Explicitly enable or disable HTTP Error logging.
- Specify the access log file path. This path can be different from the default path.
- Specify a maximum size for the error log file. This value can be larger or smaller than the default maximum size.
- Specify the type of error messages that you want included in the HTTP error log file.

Enable error logging

When selected, HTTP errors that occur while the HTTP channel processes client requests are recorded in the HTTP error log file.

Error log file path

Indicates the directory path and the name of the HTTP error log file. Standard variable substitutions, such as `$(SERVER_LOG_ROOT)`, can be used when specifying the directory path.

If you are running the product on z/OS, you should use a server-specific variable, such as `$(SERVER_LOG_ROOT)`, to prevent log file name collisions.

Error log maximum size

Indicates the maximum size, in megabytes, of the HTTP error log file. When this size is reached, the *logfile_name* archive log file is created. However, every time that the original log file overflows this archive file, this file is overwritten with the most current version of the original log file.

Maximum number of historical files

Specifies the maximum number of historical versions of the HTTP error log file that are kept for future reference.

Error log level

Specifies the type of error messages that are included in the HTTP error log file.

You can select:

Critical

Only critical failures that stop the Application Server from functioning properly are logged.

Error The errors that occur in response to clients are logged. These errors require Application Server administrator intervention if they result from server configuration settings.

Warning

Information on general errors, such as socket exceptions that occur while handling client requests, are logged. These errors do not typically require Application Server administrator intervention.

Information

The status of the various tasks that are performed while handling client requests is logged.

Debug

More verbose task status information is logged. This level of logging is not intended to replace RAS logging for debugging problems, but does provide a steady status report on the progress of individual client requests. If this level of logging is selected, you must specify a large enough log file size in the **Error log maximum size** field to contain all of the information that is logged.

TCP transport channel settings

Use this page to view and configure a TCP transport channels. This type of transport channel handles inbound TCP/IP requests from a remote client.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name > Ports**. Click on **View associated transports** for the port associated with the TCP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the TCP transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type string

Port

Specifies the TCP/IP port this transport channel uses to establish connections between a client and an application server. The TCP transport channel binds to the hostnames and ports listed for the Port property. You can specify the wildcard * (an asterisk), for the hostname if you want this channel to listen to all hosts that are available on this system. However, before specifying the wildcard value, make sure this TCP transport channel does not have to bind to a specific hostname.

Data type string

Maximum open connections

Specifies the maximum number of connections that are available for a server to use.

Leave the Maximum open connections property set to the default value 20000, which is the maximum number of connections allowed. The transport channel service by default manages high client connection counts and requires no tuning.

Default 20,000

Inactivity timeout

Specifies the amount of time, in seconds, that the TCP transport channel waits for a read or write request to complete on a socket.

If client connections are being closed without data being written back to the client, change the value specified for the Inactivity timeout parameter. This parameter controls the maximum number of connections available for a server's use. Upon receiving a new connection, the TCP transport channel waits for enough data to arrive to dispatch the connection to the protocol specific channels above the TCP transport channel. If not enough data is received during the time period specified for the Inactivity timeout parameter, the TCP transport channel closes the connection.

The default value for this parameter is 60 seconds, which is adequate for most applications. You should increase the value specified for this parameter if your workload involves a lot of connections and all of these connections can not be serviced in 60 seconds.

Note: The value specified for this property might be overridden by the wait times established for channels above this channel. For example, the wait time established for an HTTP transport channel overrides the value specified for this property for every operation except the initial read on a new socket.

Data type Integer
Default 60 seconds

Address exclude list

Lists the IP addresses that are not allowed to make inbound connections.

Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to deny access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0  
254.*.*.9  
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an Address exclude list:

```
0:>:::0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Note: The **Address include list** and **Host name include list** are processed before the **Address exclude list** and the **Host name exclude list**. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Address include list

Lists the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 or IPv6 or both addresses to which you want to grant access on inbound TCP connection requests.

All four numeric values in an IPv4 address must be represented by a number or the wildcard character * (an asterisk).

Following are examples of valid IP addresses that can be included in an Address include list:

```
*.1.255.0
254.*.*.9
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character * (an asterisk). No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number.

Following are examples of valid IPv6 addresses that can be included in an **Address include list**:

```
0:>:::0:007F:0:0001:0001
F:FF:FFF:FFFF:1:01:001:0001
1234:*:4321:*:9F9f:*:*:0000
```

Note: The Address include list and the Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it will not be allowed access.

Host name exclude list

List the host names that are not allowed to make connections. Use a comma to separate the URL addresses to which you want to deny access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, *.Rest.Of.Address. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear anywhere else in the address. For example, ibm*.com is not a valid hostname.

Following are examples of valid URL addresses that can be included in a Host name exclude list:

```
*.ibm.com
www.ibm.com
*.com
```

Note: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Host name include list

Lists the host names that are allowed to make inbound connections. Use a comma to separate the URL addresses to which you want to grant access on inbound TCP connection requests.

A URL address can start with the wildcard character * (an asterisk) followed by a period; for example, *.Rest.Of.Address. If a period does not follow the wildcard character, the asterisk will be treated as a normal non-wildcard character. The wildcard character cannot appear anywhere else in the address. For example, ibm*.com is not a valid hostname.

Following are examples of valid URL addresses that can be included in a hostname include list:

```
*.ibm.com
www.ibm.com
*.com
```

Note: The Address include list and Host name include list are processed before the Address exclude list and the Host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

DCS transport channel settings

Use this page to view and configure an DCS transport channels. This type of transport channel handles inbound Distribution and Consistency Services (DCS) messages.

By default, two channel transport chains are defined for an application server that contains a DCS channel:

- The chain named DCS contains a TCP and a DCS channel.
- The chain named DCS-Secure contains a TCP, an SSL, and a DCS channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the DCS_UNICAST_ADDRESS port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name > Ports**. Click **View associated transports** for the port associated with the DCS transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the DCS transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in the product environment. For example, a DCS transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it

might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

ORB service transport channel settings

Use this page to view and configure an Object Request Broker (ORB) Service transport channels. This type of transport channel handles CORBA and RMI/IIOP inbound messages for the ORB Service. It enables clients to make requests and receive responses from servers in a network-distributed environment.

By default, two channel transport chains are defined for an application server that contains an ORB Service channel:

- The chain named ORB contains a TCP and an ORB Service channel.
- The chain named ORB-Secure contains a TCP, an SSL, and an ORB Service channel.

Both of these chains terminate in, or use the same TCP channel instance. This TCP channel is associated with the IIOP port and is not used in any other transport chain. One instance of an SSL channel is reserved for use in the DCS-Secure chain. It also is not used in any other transport chains.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name* > Ports**. Click on **View associated transports** for the port associated with the ORB Service transport channel whose settings you want to look at.

Transport channel name

Specifies the name of the ORB service transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in the product environment. For example, an ORB service transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type	string
------------------	--------

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

SSL inbound channel

Use this page to determine which SSL inbound channel options to specify for the application server.

To view this administrative console page:

1. Click **Servers > Server Types > WebSphere application servers > *server_name***.
2. Under Container settings, click **Web container settings > Web container transport chains > *isecure_transport_chain***.
3. Under Transport channels, click **SSL Inbound Channel (SSL_1)**.

Transport Channel Name

Specifies the name of the SSL inbound channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in an application server environment. For example, an SSL inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type Positive integer

Default 0

Centrally managed

Specifies that the selection of an SSL configuration is based upon the outbound topology view for the Java Naming and Directory Interface (JNDI) platform.

Centrally managed configurations support one location to maintain SSL configurations rather than spreading them across the configuration documents.

Default: Enabled

Specific to this endpoint

Specifies the SSL configuration alias that you want to use for outbound SSL communications.

This option overrides the centrally managed configuration for the JNDI (LDAP) protocol.

Session Initiation Protocol (SIP) inbound channel settings

Use this page to configure the SIP inbound channel settings.

To view this administrative console page, click **Servers > Application servers > server_name > Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default UDP_(n) where (n) represents the number of instances of this channel in the system

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	10

Session Initiation Protocol (SIP) container inbound channel settings

Use this page to configure the SIP container inbound channel settings.

To view this administrative console page, click **Servers > Application servers > server_name > Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the SIP container inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP container transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default	UDP_(n) where (n) represents the number of instances of this channel in the system
----------------	--

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	10

Creating a new port

To create a new port and set up a channel chain to listen on a new port:

1. Go to the **Proxy Servers > SIP Proxy 1 > Transport Chain > UDP_SIP_PROXY CHAIN** panel and select **UDP inbound channel (UDP 1)**.
2. On the following panel, select the **Port** (i.e., PROXY SIP ADDRESS (*:5060)).
3. On the following panel, select **New**.

User Datagram Protocol (UDP) Inbound channel settings

Use this page to configure the UDP Inbound channel settings.

To view this administrative console page, click **Servers > Application servers > server_name > Ports** . Click on **View associated transports** for the port associated with the UDP transport channel whose settings you want to view.

Transport channel name

Specifies the name of the UDP inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' ,

This name must be unique across all channels in a WebSphere Application Server environment. For example, a UDP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Default UDP_(n) where (n) represents the number of instances of this channel in the system

Address exclude list

Specifies the IP addresses that are not allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to deny access on inbound UDP connection requests.

The address include list and host name include list are processed before the address exclude list and the host name exclude list. If all four lists are defined:

- An address that is defined on either inclusion list will be allowed access provided it is not included on either of the exclusion lists.
- If an address is included in both an inclusion list and in an exclusion list, it is not allowed access.

Data type

String

Range

Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character. All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).

Example

The following examples are valid IPv4 addresses that can be included in an Address exclude list:

```
*.1.255.0  
254.*.*.9  
1.*.*.*
```

All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*), an asterisk. No shortened version of the IPv6 address should be used. Even though a shortened version is processed with no error given, it does not function correctly in this list. Each numeric entry should be a 1- 4 digit hexadecimal number. The following examples are valid IPv6 addresses that can be included in an Address exclude list:

```
0:*:*:0:007F:0:0001:0001  
F:FF:FFF:FFFF:1:01:001:0001  
1234:*:4321:*:9F9f:*:*:0000
```

Address include list

Specifies the IP addresses that are allowed to make inbound connections. Use a comma to separate the IPv4 and/or IPv6 addresses to which you want to allow access on inbound UDP connection requests.

Data type

String

Range

Valid IPv4 and IPv6 addresses with a wildcard character (*), an asterisk. All four elements of an IPv4 address must be represented by a number or a wildcard character (*). All eight numeric values of an IPv6 address must be represented by a number or the wildcard character (*).

Web container inbound transport channel settings

Use this page to view and configure a Web container inbound channel transport. This type of channel transport handles inbound Web container requests from a remote client.

To view this administrative console page, click **Servers** → **Server Types** → **WebSphere application servers** → *server_name* → **Web Container Settings** → **Web container** → **Web container transport chains** → *transport_chain* → **Web container inbound channel** (*transport_channel_name*).

Transport Channel Name

Specifies the name of the Web container inbound transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, a Web container inbound transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type

String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type

Positive integer

Default

0

Write buffer size

Specifies the amount of content in bytes to buffer unless the servlet explicitly calls flush/close on the response/writer output stream.

Data type

bytes

Default

32768 bytes

DataPower appliance manager transport channel settings

Use this page to view and configure a DataPower® appliance manager transport channel. This type of transport channel handles events from managed DataPower appliances.

To view this administrative console page, click **System administration . Deployment manager > Ports**. Find DataPowerMgr_inbound_secure, in the list of channels, and click **View associated transports**. In the list of associated transports, click DataPowerManagerInboundSecure, and then, under Transport Channels, click **DataPower appliance manager inbound channel (DPMGRDPMGR_1)**.

Transport channel name

Specifies the name of the transport channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP transport channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the transport chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type Positive integer

Default 0

HTTP transport channel custom properties

If you are using an HTTP transport channel, you can add any of the following custom properties to the configuration settings for that channel.

To add a custom property:

1. In the administrative console, click **Servers > Server Types**, and then select one of the following options, depending on the type of chain you are creating:
 - **Application servers > server_name, > Web container settings > Web container transport chains > chain_name > HTTP Inbound Channel > Custom Properties > New.**
 - **Proxy servers**, and then under HTTP Proxy Server Settings, click **Proxy server transports**. Then, select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**, and then click **> HTTP Inbound Channel > Custom Properties > New.**

Note: There are four Web container transport chains:

WCInboundAdmin
WCInboundAdminSecure
WCInboundDefault
WCInboundDefaultSecure

An application server inherits the custom property values that are specified for the WCInboundAdmin or WCInboundAdminSecure transport chain because one of these chains is usually the first chain to get activated when the application server is initialized. Therefore, before specifying any custom properties for a Web container transport chain, you should disable the WCInboundAdmin and WCInboundAdminSecure transport chains.

2. Under **General Properties** specify the name of the custom property in the Name field and a value for this property in the Value field. You can also specify a description of this property in the Description field.
3. Click **Apply** or **OK**.
4. Click **Save** to save your configuration changes.
5. Restart the server.

Following are the descriptions of the HTTP transport channel custom properties provided with the product. These properties are not shown on the settings page for an HTTP transport channel.

CookiesConfigureNoCache

Use the CookiesConfigureNoCache property to specify whether the presence of a Set-Cookie header in an HTTP response message triggers the addition of several cache related headers. If this property is set to true, an Expires header with a very old date, and a Cache-Control header that explicitly tells the client not to cache the Set-Cookie header are automatically added. These headers are not automatically added if this property is set to false.

Data type	Boolean
Default	True

localLogFilenamePrefix

Use the localLogFilenamePrefix property to specify a prefix for the filename of the network log file. Normally, when inprocess optimization is enabled, requests through the inprocess path are logged based on the logging attributes set up for the Web container's network channel chain. You can use this property to add a prefix to the filename of the network log file. This new filename is then used as the filename for the log file for inprocess requests. Requests sent through the inprocess path are logged to this file instead of to the network log file. For example, if the log file for a network transport chain is named ../httpaccess.log, and this property is set to local for the HTTP channel in that chain, the filename of the log file for inprocess requests to the host associated with that chain is ../localhttpaccess.log.

Note: If you specify a value for the localLogFilenamePrefix custom property, you must also set the accessLogFileName HTTP channel custom property to the fully qualified name of the log file you want to use for in process requests. You cannot specify a variable, such as \$(SERVER_LOG_ROOT), as the value for this custom property.

Data type	String
------------------	--------

limitFieldSize

Use the limitFieldSize property to enforce the size limits on various HTTP fields, such as request URLs, or individual header names or values. Enforcing the size limits of these fields guards against possible Denial of Service attacks. An error is returned to the remote client if a field exceeds the allowed size.

Data type	Integer
Default	32768
Range	50-32768

limitNumHeaders

Use the limitNumHeaders property to limit the number of HTTP headers that can be present in an incoming message. If this limit is exceeded, an error is returned to the client.

Data type	Integer
Default	500
Range	50 to 500

RemoveServerHeader

Use the RemoveServerHeader property to force the removal of any server header from HTTP responses that the application server sends, thereby hiding the identity of the server program.

Data type	Boolean
Default	False

ServerHeaderValue

Use the ServerHeaderValue property to specify a header that is added to all outgoing HTTP responses if a server header does not already exist.

Data type	String
Default	WebSphere Application Server v/x.x, where x.x is the version of WebSphere Application Server that is running on your system.

HTTP Tunnel transport channel custom property

If you are using an HTTP Tunnel transport channel, you can add the following custom property to the configuration settings for that channel.

To add a custom property:

1. In the administrative console, click **Servers > Server Types > Application servers > *server_name* > Ports**. Click on **View associated transports** for the HTTP Tunnel port to whose configuration settings you want to add this custom property.
2. Click **New**.
3. Under **General Properties** specify the name of the custom property in the Name field and a value for this property in the Value field. You can also specify a description of this property in the Description field.
4. Click **Apply** or **OK**.
5. Click **Save** to save your configuration changes.
6. Restart the server.

Following is a description of the HTTP Tunnel transport channel custom property that is provided with the product. This property is not shown on the settings page for an HTTP Tunnel transport channel.

pluginConfigurable

Indicates whether or not the configuration settings for the HTTP Tunnel transport channel are included in the plugin-cfg.xml file for the Web server associated with the application server that is using this channel.

Configuration settings for each of the Web container transport channels defined for an application server are automatically included in the plugin-cfg.xml file for the Web server associated with that application server.

Data type	Boolean
Default	False

TCP transport channel custom properties

If you are using a TCP transport channel, you can use TCP transport channel custom properties to configure internal TCP transport channel properties.

To add a TCP transport channel custom property, perform the following actions.

1. In the administrative console, click **Servers > Server Types**, and then follow one of the following paths:
 - **Application servers > *server_name***, and then select one of the following options, depending on the type of chain you are creating:
 - Expand **SIP container settings**, and click **SIP container transport chains**.
 - Expand **Web container settings**, and click **Web container transport chains**.
 - Under **Container services**, and click **ORB Service > ORB Service Transport Chains**.

- Expand **Server messaging**, and click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.
 - **Proxy servers**, and then expand **HTTP proxy server settings**, and click **Proxy server transports** and select either **HTTPS_PROXY_CHAIN** or **HTTP_PROXY_CHAIN**. Then click **HTTP proxy inbound channel**
2. Select the transport chain that includes the TCP channel for which you want to specify the custom property.
 3. Select the **TCP inbound channel**.
 4. Click **Custom properties > New**, expand **General properties**, and specify the name of the custom property in the **Name** field and a value for this property in the **Value** field. You can also specify a description of this property in the **Description** field.
 5. Click **Apply** or **OK**.
 6. Click **Save** to save your configuration changes.
 7. Restart the server.

The following TCP transport channel custom properties are provided with the product. These properties are not shown on the settings page for a TCP transport channel.

listenBacklog

Use the listenBacklog property to specify the maximum number of outstanding connection requests that the operating system can buffer while it waits for the application server to accept the connections. If a client attempts to connect when this operating system buffer is full, the connect request is rejected. The value of this property is specific to each transport.

If you need to control the number of concurrent connections, use the **Maximum open connections** field on the administrative console TCP transport channel settings page.

Data type	Integer
Default	511

zaiofreeInitialBuffers

Use the zaiofreeInitialBuffers property to indicate that the TCP channel should release the initial read buffers used on new connections as soon as these buffers are no longer needed for the connection. By default, this initial read buffer is cached for each connection. When a connection is closed, the read buffer is reused to avoid a memory allocation. This process works well for non-persistent connections, where there is one request per connection. However, for highly persistent connections, the buffer might be held for a considerable amount of time even though it is not being used. For workloads that require a large number of connected clients, this situation can cause a shortage of Linkage Editor (LE) heap space. Unless your workload consists mainly of non-persistent connections, you should set this custom property to true to enable the release of the initial read buffers.

Note: If you set this property to true, you must also add the following argument to the JVM generic arguments that are configured for the application server that is using this TCP channel:

```
-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=
com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl
```

Data type	String
Default	false

Web container transport channel custom properties

Use this page to set custom properties for a Web container transport channel.

To specify custom properties for a specific transport on the Web container transport chain:

1. In the administrative console click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Web Container Settings** → **Web container transport chains**.
2. Select a transport chain.
3. Under **Transport Channels** select **Web container inbound channel (channel_name)**.
4. Under **Additional Properties** select **Custom properties**.
5. On the Custom properties page, click **New**.
6. On the settings page, enter the property that you want to configure in the **Name** field and the value that you want to set it to in the **Value** field.
7. Click **Apply** or **OK**.
8. Click **Save** on the console task bar to save your configuration changes.
9. Restart the server.

Following is a list of custom properties provided with the Application Server. These properties are not shown on the settings page for a Web container transport.

disableRequestMessageChunking

This custom property disables request message chunking when set to *true*. All of the request body up to `protocol_http_large_data_inbound_buffer` is buffered in memory.

For `WCInboundAdmin` and `WCInboundAdminSecure` transport chains, chunking is enabled by default to install large EAR files through the administrative console. For example, the settings for these chains are `disableRequestMessageChunking=false`. When chunking is enabled, the `protocol_http_large_data_inbound_buffer` value is ignored because the entire HTTP request is not buffered in the controller.

When chunking is disabled, the `protocol_http_large_data_inbound_buffer` value is used because the entire HTTP request is buffered in the controller.

Property name	<code>disableRequestMessageChunking</code>
Data type	string
Value	True or False
Defaults	By default, administrative chains have the <code>disableRequestMessageChunking</code> custom property explicitly set to true.

maxRequestMessageBodySize

When `disableRequestMessageChunking` is set to false, this is the maximum amount of request body that is buffered in memory before sending the next chunk to the servant. The `maxRequestMessageBodySize` custom property is valid only if the `disableRequestMessageChunking` custom property is set to false.

Property name	<code>maxRequestMessageBodySize</code>
Value	The default value is 32 kilobytes (KB). The minimum value is 32 and the maximum value is 8192, which is equivalent to 8MB.

Configuring inbound HTTP request chunking

Inbound HTTP request chunking is used to eliminate the restriction on messages greater than 10MB. The 10MB restriction is set because the entire message is buffered in the controller before the HTTP request is dispatched to the servant, therefore, the controller may fail with an out of memory condition when multiple large HTTP messages are processed simultaneously. With chunking enabled, the message is broken up into smaller pieces before it is processed by the Web container and application. As a result, only one small

chunk is buffered in memory at a time in the controller thus greatly reducing the amount of memory consumed by large HTTP messages. Applications do not require changes to enable inbound HTTP chunking.

About this task

Inbound HTTP request chunking, is configured at the Web container transport chain level. You can configure each Web container chain to enable or disable chunking. When chunking is enabled for a particular chain, you can also configure the maximum chunk size for chunking enabled for each chain.

All HTTP Web container chains have chunking enabled by default.

1. In the administrative console click **Servers** → **Server Types** → **WebSphere application servers** → **server_name** → **Web Container Settings** → **Web container transport chains**.
2. Select a transport chain.
3. Under **Transport Channels** select **Web container inbound channel (channel_name)**.
4. Under **Additional Properties** select **Custom properties** to configure inbound HTTP request message chunking. See the article, “Web container transport channel custom properties” on page 254 for details about request message chunking settings.
 - a. If the **disableRequestMessageChunking** property is already defined, select the **disableRequestMessageChunking** property from the list.
 - b. If the **disableRequestMessageChunking** property is not defined, click **new**.
5. On the settings page, do one of the following:
 - To enable request message chunking, enter the property, **disableRequestMessageChunking** in the **Name** field and the enter the value, false, in the **Value** field. Click **Apply** or **OK** so save the custom property changes.
 - To disable request message chunking, enter the property, **disableRequestMessageChunking** in the **Name** field and the enter the value, true, in the **Value** field. Click **Apply** or **OK** so save the custom property changes.
6. Configure message chunk size if request message chunking is enabled. See the article, “Web container transport channel custom properties” on page 254 for details on these settings.
 - a. On the Custom Properties page, click **New**.
 - b. On the settings page, enter the property, **maxRequestMessageBodySize**, in the Name field and the enter a size, specified in kilobytes, between 32 and 8192 in the Value field.
 - c. Click **Apply** or **OK**.
7. Click **Save** on the console task bar to save your configuration changes.
8. Restart the server.

Transport chain problems

Review the following topics if you encounter a transport chain problem.

TCP transport channel fails to bind to a specific host/port combination

If a TCP transport channel fails to bind to a specific port, one of the following situations might have occurred:

- You are trying to bind the channel to a port that is already bound to another application, such as another instance of an application server.
- You are trying to bind to a port that is in a transitional state waiting for closure. This socket must transition to closed before you restart the server. The port might be in TIME_WAIT, FIN_WAIT_2, or CLOSE_WAIT state. Issue the `netstat -a` command from a command prompt to display the state of the port to which you are trying to bind.

Deleting a transport chain

Transport chains cannot be deleted the same way that HTTP transports can be deleted. Because you cannot have multiple HTTP transports associated with the same port, when you delete an HTTP transport, you effectively delete the associated port and stop all traffic on that port. However, the process is more complicated for a transport chain because multiple transport chains might be associated with the same port and you do not want to disrupt traffic on transport chains that you are not deleting.

Before you begin

Determine whether you want to delete a particular transport chain or all of the transport chains that are associated with a specific port.

About this task

You might have to delete one or more transport chains if you have to delete a port.

To delete a transport chain:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.
3. Select the transport chain you want to delete, and click **Delete**. If you intend to delete the port that is associated with this transport chain, repeat this step for all of the transport chains associated with this port.
4. Click **Save** to save your changes.

What to do next

If you delete all of the transport chains associated with a port, you can delete the port.

Disabling ports and their associated transport chains

Transport chains cannot be disabled the same way that HTTP transports can be disabled. Because you cannot have multiple HTTP transports associated with the same port, when you disable an HTTP transport, you effectively disable the associated port and stop all traffic on that port. However, the process is more complicated for a port that has associated transport chains because multiple transport chains might be associated with the same port, and you might not want to disrupt traffic on all of the transport chains at the same time.

Before you begin

Determine whether you want to disable a particular transport chain or all of the transport chains that are associated with a specific port.

About this task

You might need to disable a transport chain if you want to temporarily stop all incoming traffic on a particular port or on a particular transport chain that is associated with that port.

To disable a specific transport chain:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name* > Ports**.
2. In the list of available ports, locate the port that you want to delete and click **View associated transports** for that port.

3. Click the transport chain you want to disable.
4. Unselect the **Enabled** field, and click **OK**. If you want to temporarily stop all of the incoming traffic on a port, repeat this step for all of the transport chains associated with this port.
5. Click **Save** to save your changes.

What to do next

When you want traffic to resume on these disabled transport chains, repeat the preceding steps for all of the transport chains you disabled, and select the **Enabled** field.

Creating custom services

You can create one or more custom services for an application server. Each custom services defines a class that is loaded and initialized whenever the server starts and shuts down. Each of these classes must implement the `com.ibm.websphere.runtime.CustomService` interface. After you create a custom service, use the administrative console to configure that custom service for your application servers.

About this task

Custom services run in servants, not in controllers. For example, because there can be more than one servant started in the life of a server and these servants can be started long after the server (controller) is up, as needed by WLM, a custom service runs during the start of each servant.

If you need to define a hook point that runs when a server starts and shuts down, create a custom service class and then use the administrative console to configure a custom service instance. When the application server starts, the custom service starts and initializes.

Following is a list of restrictions that apply to the product custom services implementation. Most of these restrictions apply only to the initialize method:

- The initialize and shutdown methods must return control to the runtime.
- No work is dispatched into the server instance until all custom service initialize methods return.
- The initialize and shutdown methods are called only once on each service, and once for each operating system process that makes up the server instance.
- Initialization of process level static data, without leaving the process, is supported.
- Only JDBC RMLT (resource manager local transaction) operations are supported. Every unit of work (UOW) must be completed before the methods return.
- Creation of threads is not supported.
- Creation of sockets and I/O, other than file I/O, is not supported.
- Running standard Java Platform, Enterprise Edition (Java EE) code, such as client code, servlets, and enterprise beans, is not supported.
- The Java Transaction API (JTA) interface is not available.
- This feature is available in Java EE server processes and distributed generic server processes only.
- While the runtime makes an effort to call shutdown, there is no guarantee that shutdown will be called prior to process termination.
- JNDI operations that request resources are not supported.

1. Develop a custom service class that implements the `com.ibm.websphere.runtime.CustomService` interface.

The `com.ibm.websphere.runtime.CustomService` interface includes an initialize and shutdown methods. The application server uses the initialize method to pass properties to the custom service. These properties can include:

- A property that provides the name of an external file that contains configuration information for the service. You can use the `externalConfigURLKey` property to retrieve this information.

- Properties that contain name-value pairs that are stored for the service, along with the other system administration configuration data for the service.

Both the initialize and shutdown methods declare that they might create an exception, although no specific exception subclass is defined. If either method creates an exception, the runtime logs the exception, disables the custom service, and continues to start the server.

2. Configure the custom service.

In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***, and then under Server Infrastructure, click **Custom Services > New**. Then, on the settings page for a custom service instance, create a custom service configuration for an existing application server or node agent, supplying the name of the class implemented. If your custom service class requires a configuration file, specify the fully-qualified path name to that configuration file in the **externalConfigURL** field. This file name is passed into your custom service class.

To invoke a native library from the custom service, provide the path name in the **Classpath** field in addition to the path names that are used to locate the classes and JAR files for the custom service. This procedure adds the path name to the extension classloader, which allows the custom service to locate and correctly load the native library.

3. Stop the application server, and then restart it.

Stop the application server, and then restart the server.

Results

Each custom services defines a class that is loaded and initialized whenever the server starts and shuts down.

The custom service loads and initializes whenever the server starts and shuts down.

Example

As previously mentioned, your custom services class must implement the `com.ibm.websphere.runtime.CustomService` interface. In addition, your class must implement the initialize and shutdown methods. The following example, shows the code that declares the class `ServerInit` that implements your custom service. This code assumes that your custom service class needs a configuration file. This example also includes the code that accesses the external configuration file. If your class does not require a configuration file, you do not have to include the `configProperties` portion of this code.

```
public class ServerInit implements com.ibm.websphere.runtime.CustomService
{
/**
 * The initialize method is called by the application server runtime when the
 * server starts. The Properties object that the application server passes
 * to this method must contain all of the configuration information that this
 * service needs to initialize properly.
 *
 * @param configProperties java.util.Properties
 */
    static final java.lang.String externalConfigURLKey =
        "com.ibm.websphere.runtime.CustomService.externalConfigURLKey";

    static String ConfigFileName="";

    public void initialize(java.util.Properties configProperties) throws Exception
    {
        if (configProperties.getProperty(externalConfigURLKey) != null)
        {
            ConfigFileName = configProperties.getProperty(externalConfigURLKey);
        }

        // Implement rest of initialize method
    }
}
```



```

/**
 * The shutdown method is called by the application server runtime when the
 * server begins its shutdown processing.
 *
 * public void shutdown() throws Exception
 * {
 *     // Implement shutdown method
 * }

```

What to do next

Check the application server to verify that the initialize and shutdown methods of the custom service run the way that you want them to run.

Custom service collection

Use this page to view a list of services available to the application server and to see whether the services are enabled. A custom service provides the ability to plug into an application server and define code that runs when the server starts or shuts down.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, in the Server Infrastructure section, click **Administration > Custom services**.

External Configuration URL

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, the value provides a fully-qualified path name to that configuration file. This file name is passed into your custom service class.

Classname

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Display Name

Specifies the name of the service.

Enable service at server startup

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Custom service settings

Use this page to configure a service that runs in an application server.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, in the Server Infrastructure section, click **Administration > Custom services > *custom_service_name***.

Enable service at server startup:

Specifies whether the server attempts to start and initialize the service when its containing process (the server) starts. By default, the service is not enabled when its containing process starts.

Data type	Boolean
Default	false

External Configuration URL:

Specifies the URL for a custom service configuration file.

If your custom services class requires a configuration file, specify the fully-qualified path name to that configuration file for the value. This file name is passed into your custom service class.

Data type	String
Units	URL

Classname:

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

Data type	String
Units	Java class name

Display Name:

Specifies the name of the service.

Data type	String
------------------	--------

Description:

Describes the custom service.

Data type	String
------------------	--------

Classpath:

Specifies the class path used to locate the classes and JAR files for this service.

Data type	String
Units	Class path

Defining application server processes

To enhance the operation of an application server, you can define command-line information for starting or initializing an application server process. Such settings define runtime properties such as the program to run, arguments to run the program, and the working directory.

About this task

A process definition can include characteristics such as Java virtual machine (JVM) settings, standard in, error and output paths, and the user ID and password under which a server runs.

You can define application server processes using the administrative console or the wsadmin tool.

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and then click on an application server name.
2. In the Server Infrastructure section, click **Java and process management > Process definition**.
3. Select either **Control**, **Servant**, or **Adjunct**.

4. On the settings page for a process definition, specify the name of the executable to run, any arguments to pass when the process starts running, and the working directory in which the process will run. Then click **OK**.
5. Specify process execution statements for starting or initializing a UNIX or i5/OS process.
6. Specify monitoring policies to track the performance of a process.
7. Specify process logs to which standard out and standard error streams write. Complete this step if you do not want to use the default file names.
8. Specify name-value pairs for properties needed by the process definition.

Note: Each custom property name must be unique. If the same name is used for multiple properties, the process uses the value specified for the first property that has that name.

9. Stop the application server, and then have the executable, that the process definition specifies, restart the server. If the executable cannot restart the application server, the executable should use the generic server.
10. Check the server to verify that the process definition runs and operates as intended.

Process definition settings

Use this page to configure a process definition. A process definition includes the command line information necessary to start or initialize a process.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, in the Server Infrastructure section, click **Java and process management > Process definition**.

On z/OS you must then click **Control, Servant, or Adjunct**.

For z/OS, this page provides command-line information for starting, initializing, or stopping a process. Each of the commands for which information is provided can be used for the control process. Only the Start command and Start command arguments properties apply for the servant process. Specify the commands for the control process on one process definition panel and the commands for the servant process on another process definition panel. Do not specify the commands for the two different processes on the same panel.

Start command (`startCommand`)

This command line information specifies the platform-specific command to launch the server process.

z/OS control process

Data type	String
Format	START <i>control_JCL_procedure_name</i>
Example	START BBO6ACR

z/OS servant process

For the z/OS servant process, the value on the start command specifies the procedure name that workload manager (WLM) uses to start the servant process. WLM only uses this value if the WLM dynamic application environment feature is installed.

Data type	String
Format	<i>servant_JCL_procedure_name</i>
Example	BBO6ASR

Start command arguments (`startCommandArgs`)

This command line information specifies any additional arguments required by the start command.

z/OS control process

Data type	String
Format	JOBNAME= <i>server_short_name</i> , ENV= <i>cell_short_name.node_short_name.server_short_name</i>
Example	JOBNAME=BBOS001,ENV=SY1.SY1.BBOS001

z/OS servant process

Data type	String
Format	JOBNAME= <i>server_short_name</i> S, ENV= <i>cell_short_name.node_short_name.server_short_name</i>
Example	JOBNAME=BBOS001S,ENV=SY1.SY1.BBOS001

Note: For z/OS, the server short name (JOBNAME) contains 7 characters by default, but you can lengthen the short name to 8 characters.

Stop command (stopCommand)

This command line information specifies the platform-specific command to stop the server process

For z/OS, if this field is left blank, then the MVS STOP command is used to stop the generic server.

Specify two commands in the field, one for the Stop command, and one for the Immediate Stop (CANCEL) command.

Data type	String
Format	STOP <i>server_short_name</i> ;CANCEL <i>server_short_name</i>
z/OS example	STOP BBOS001;CANCEL BBOS001

Stop command arguments (stopCommandArgs)

This command line information specifies any additional arguments required by the stop command.

Specify arguments for the Stop command and the Immediate Stop (CANCEL) command.

Data type	String
Format	<i>stop command arg string</i> ; <i>immediate stop command arg string</i>
z/OS Example	;ARMRESTART

In this example, Stop has no arguments. Immediate Stop has the argument ARMRESTART. A semicolon precedes ARMRESTART.

Terminate command (terminateCommand)

This command line information specifies the platform-specific command to terminate the server process.

Data type	String
Format	FORCE <i>server_short_name</i>
z/OS example	FORCE BBOS001

Terminate command arguments (terminateCommandArgs)

This command line information specifies any additional arguments required by the terminate command.

The default is an empty string.

Data type	String
Format	<i>terminate command arg string</i>
z/OS example	ARMRESTART

Working directory

Specifies the file system directory that the process uses as its current working directory. This setting only applies for i5/OS and distributed platforms. The process uses this directory to determine the locations of input and output files with relative path names.

This field does not display for the z/OS control process.

Note: On z/OS, the working directory is always the UNIX System Services directory that is defined using the OMVS setting of the RACF user profile for the user that starts the servant. Therefore, even if you specify a directory in this field, the UNIX System Services directory is used as the working directory. To provide compatibility between applications that run on a z/OS platform and on a distributed platform, set the UNIX System Services directory to the same value that you specify for the **Working directory** field on your distributed platform system.

Data type	String
------------------	--------

Executable target type

Specifies whether the executable target is a Java class or an executable JAR file.

Executable target

Specifies the name of the executable target. If the target type is a Java class name, this field contains the main() method. If the target type is an executable JAR file, this field contains the name of that JAR file.

Data type	String
------------------	--------

Process execution settings

Use this page to view or change the process execution settings for a server process.

A server process applies to either an application server, a node agent or a deployment manager.

If you are running on z/OS, to view this administrative console page for an application server, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, in the Server Infrastructure section, click **Java and process management**, click either **Servant**, **Control** or **Adjunct**, and then click **Process execution**.

To view this administrative console page for a node agent, click **System Administration > Node agents > *nodeagent_name***. Then, under Server Infrastructure, click **Java and process management > Process definition > Process execution**.

To view this administrative console page for a deployment manager, click **System Administration > Deployment manager**. Then, in the Server Infrastructure section, click **Java and process management**, click either **Servant**, **Control** or **Adjunct**, and then click **Process execution**.

Process Priority:

Specifies the operating system priority for the process. The administrative process that launches the server must have root operating system authority in order to honor this setting.

Data type	Integer
------------------	---------

Default

20

UMASK:

Specifies the user mask under which the process runs (the file-mode permission mask).

The deployment manager and application servers must run with a 007 umask in order to support system management functions. Therefore, it is recommended that you do not change the default value of this setting for the deployment manager or the controller.

If the process is running in a servant, you can either specify a different user mask setting in this field or use the `_EDC_UMASK_DFLT` environment variable in the JCL procedure for the servant to change this setting.

Data type	Integer
Default	007

Run As User:

Specifies the user that the process runs as. This user ID must be defined to the security system.

Data type	String
------------------	--------

Run As Group:

Specifies the group that the process is a member of and runs as.

Data type	String
------------------	--------

Run In Process Group:

Specifies a specific process group for the process. A process group is a mechanism that the operating system uses to logically associate multiple processes and operate on them as a single unit. Usually, the operating system uses this mechanism for signal distribution.

Specific operating systems might allow other operations to be performed on a process group. Refer to your operating system documentation for more information on the operations that can be performed on a process group.

Data type	Integer
Default	0, which indicates that the process is not assigned to a specific process group.

Monitoring policy settings

Use this page to view or change settings that control how the node agent monitors and restarts a process.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, under Server Infrastructure, click **Java and process management > Monitoring policy**.

Maximum Startup Attempts:

Specifies the maximum number of times to attempt to start the application server before giving up.

Data type Integer

Ping Interval:

Specifies, in seconds the frequency of communication attempts between the parent process, such as the node agent, and the process it has spawned, such as an application server. Adjust this value based on your requirements for restarting failed servers. Decreasing the value detects failures sooner; increasing the value reduces the frequency of pings, reducing system overhead.

Data type	Integer
Range	Set the value greater than or equal to 0 (zero) and less than 2147483. If you specify a value greater than 2147483, the application server acts as though you set the value to 0. When you specify a value of 0, no checking is performed.

In a z/OS environment, the Ping Interval setting for a deployment manager or a node agent is ignored. However, the Ping Interval setting for an application server is used by the node agent to control the native z/OS operating system PidWaiter monitoring function. PidWaiter monitoring is similar in functionality to the pinging function that is used in a distributed platform environment. Both of these monitoring functions determine whether an application server is still running. The only difference between the two monitoring functions is that PidWaiter monitoring does not send any of the TCP/IP messages that Ping Interval monitoring sends.

You can also set the following two properties to considerably reduce the number of DNS lookups that might occur because of this monitoring activity:

1. You can add the JVM custom property `com.ibm.websphere.management.monitoring.pingInterval` for the controller for each process. The default value for this property is 60 seconds. It is not recommended that you change this default value unless you need to minimize the number of DNS lookups that occur. If you need to minimize the number of DNS lookups that occur, set this property to a time interval that is more appropriate for your system.

When this property is set for the deployment manager, it regulates how frequently the deployment manager checks to see if the node agent is still running. When it is set for the node agent, it regulates how frequently the node agent checks to see if the deployment manager is still running. When it is set for an application server, it regulates how frequently the application server checks to see if the node agent is still running.

2. You can add the environment variable `protocol_iiop_resolve_foreign_hostname` at the cell level, and set to 0. Setting this variable to 0 disables the IIOP resolve foreign hostname function, thereby eliminating the DNS lookups this function performs.

Adding these two properties does not completely eliminate DNS lookups from within product processes.

Ping Timeout:

When a parent process is spawning a child process, such as when a process manager spawns a server, the parent process pings the child process to see whether the child was spawned successfully. This value specifies the number of seconds that the parent process should wait (after pinging the child process) before assuming that the child process failed.

Data type	Integer
Units	Seconds

Range Set the value greater than or equal to 0 (zero) and less than 2147483647. If you specify a value greater than 2147483647, the application server acts as though you set the value to 0.

Automatic Restart:

Specifies whether the process should restart automatically if it fails.

If you change the value specified for this field, you must restart the application server and the node agent before the new setting takes effect.

This setting does not affect what you specified for the Node Restart State setting. The two settings are mutually exclusive.

Data type Boolean
Default false for the z/OS environment

Node Restart State:

The setting only displays for the Network Deployment product. It specifies the desired behavior of the servers after the node completely shuts down and restarts.

If a server is already running when the node agent stops, that server is still running after the node agent restarts. If a server is stopped when the node agent restarts, whether the node agent starts the server depends on the setting for this property:

- If this property is set to STOPPED, node agent does not start the server.
- If this property is set to RUNNING, the node agent always starts the server.
- If this property is set to PREVIOUS, the node agent starts the server only if the server was running when the node agent stopped.

This setting does not affect what you specified for the Automatic Restart setting. The two settings are mutually exclusive.

Data type String
Default STOPPED
Range Valid values are STOPPED, RUNNING, or PREVIOUS. If you want the process to return to its current state after the node restarts, use PREVIOUS.

Process definition type settings

Use this page to view or change settings for a process definition type. This page only displays if you are running the product on z/OS.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name**. Then, under Server Infrastructure, click **Java and process management > Process definition**.

Control:

Specifies the process definitions for the control process

Servant:

Specifies the process definitions for the servant process.

Sysplex Distributor

The IBM-recommended implementation, if you are running in a sysplex, is to set up your TCP/IP network with Sysplex Distributor. This makes use of dynamic virtual IP addresses (DVIPAs), which increase availability and aid in workload balancing.

The following are recommended environment considerations for Sysplex Distributor:

- You need only basic sysplex functionality to utilize DVIPAs and Sysplex Distributor because these functions do not rely on data stored permanently in the coupling facility.
- Set up your system such that each HTTP request connection results in no saved state or the HTTP and application servers are configured to share a persistent state.

When doing this, HTTP server plug-ins send no-affinity connections to Sysplex Distributor (a secondary connection load balancer) with more information to make a better distribution decision.

Note: As long as the HTTP catcher itself is not bound to any particular IP address, the application-specific DVIPA can be used when affinities dictate a particular server. This allows use of the Sysplex Distributor server address for requests that are not tied to a server, covering the same set of servers in the sysplex.

Since the client connection terminates at the plug-in/proxy, and the secondary connection is established by the plug-in itself, there is no need for network address translation.

Requests to the node agent do not require any affinity, and each request is independent of other requests. Sysplex Distributor can be used to balance work requests among node agents, with the added benefit that Sysplex Distributor knows which nodes are available. Therefore, it will never route a work request to a node that is not listening for new connection requests.

Note: If you are running z/OS Version 1.2 or earlier, Sysplex Distributor is limited to distribution on only four ports for a particular distributed DVIPA. You may configure multiple DVIPAs when more than four ports exist, but this is a configuration burden.

Running multiple TCP/IP stacks

You might want to run multiple TCP/IP stacks on the same system to provide network isolation for one or more of your applications. For instance, you may have multiple overflow sequential access (OSA) features, each one connecting your system to a different network. You can assign a TCP/IP stack to each feature.

Before you begin

When configuring the product on a system with multiple stacks, you must first establish the product's stack affinity to the desired stack. Establishing stack affinity binds all socket communications to that stack, and allocates the proper host name resolution configuration data sets to the product. These data sets enable host name lookups to have the desired results.

Use the NETWORK DOMAINNAME parameter of SYS1.PARMLIB(BPXPRMxx) to specify the common INET physical file system, C_INET PFS, and then use this file system to set up multiple TCP/IP stacks. This physical file system allows you to configure multiple physical file systems (network sockets) and make them active concurrently.

If you plan to configure the product to use a non-default TCP/IP stack, consult *z/OS UNIX System Services Planning*, and *z/OS Communications Server: IP Configuration Reference*, for details.

About this task

Note: In the steps below, you will set a number variables. It is important to understand that these variables should be set at the node level.

To configure the product on a system with multiple stacks:

1. Configure the data set for each application server's host name resolution. In the administrative console, click **Environment > WebSphere variables > New**.
 - a. Add the RESOLVER_CONFIG UNIX process variable and specify the data set name in the **value** field.
 - b. Export the RESOLVER_CONFIG variable in client shell scripts.
 - You can also use JCL to specify the name resolution configuration data set. To use JCL, add //SYSTCPD DD DSN=some.tcpip.DATA,DISP=SHR to the server JCL. The RESOLVER_CONFIG variable overrides the SYSTCPD DD statement.

See *z/OS Communications Server: IP Configuration Reference*, for more information on the RESOLVER_CONFIG variable.

2. Establish the Application Server's stack affinity to the desired stack.
 - a. In the administrative console, click **Environment > WebSphere variables** and set the _BPXK_SETIBMOPT_TRANSPORT UNIX process variable to the value of the desired transport. If this variable does not exist, click **New** and add it.
 - b. Export the _BPXK_SETIBMOPT_TRANSPORT variable in client shell scripts.

To set the **BPXK_SETIBMOPT_TRANSPORT** variable in the was.env file for the Daemon, you must prefix the variable with **DAEMON_**. This additional information causes the transformer that generates the was.env files to put add the variable to the was.env file for the Daemon. Because the **_BPXK_SETIBMOPT_TRANSPORT** variable already has a leading underscore, the final version of this variable, when set for the Daemon, contains two underscores preceding the variable name, as shown here **DAEMON__BPXK_SETIBMOPT_TRANSPORT**.

Note: If you are setting this variable for the Daemon, you probably want to set it at the cell level to give all the Daemons in that cell the same setting. Unless one of the Daemons is serving multiple nodes, if for some reason you need to specify different settings for different Daemons in a cell, , you can set this variable at the node level.

See *z/OS UNIX System Services Planning*, for more information on the _BPXK_SETIBMOPT_TRANSPORT variable.

Bind-specific support

Bind-specific support enables you to control the use of the product TCP/IP resources.

This support allows you to have the Application Server ORB and other products and applications on the same z/OS system without requiring the client code to configure unique ports. In other words, this support allows use of port 2809 by the Application Server and other products and applications on the same system. This support allows the utilization of multiple TCP/IP stacks (Common INET) by the product ORB and the use of multiple IP addresses on the same TCP/IP stack.

To use bind-specific support, use the **ORB_LISTENER_ADDRESS** end point, which specifies the IP address in dotted decimal format. The application servers listen for client connection requests on this IP address.

Because a given IP address is associated with a given TCP/IP stack, you can specify the ORB_LISTENER_ADDRESS endpoint so that the applications servers use a specific TCP/IP stack.

In addition, because you can define multiple IP addresses for a given TCP/IP stack, the Application Server port 2809 servers could share the same TCP/IP stack with other products and applications requiring port 2809, because you made their IP addresses unique with the ORB_LISTENER_ADDRESS end point.

Alternatively, you can, without the use of bind-specific support, define alternate ports for port 2809 and the location service daemon, which are the only values defined by the CORBA standard. However it is not clear that all client ORBs will easily support configuring the Application Server port to something other than 2809. Configure the ports for the location service daemon and node by specifying port numbers on the z/OS location service daemon settings page in the administrative console.

For more information about multiple TCP/IP stacks (Common INET), see *z/OS UNIX System Services Planning*. For more information about multiple IP addresses on the same TCP/IP stack, see *z/OS Communications Server: IP Configuration Reference*.

Configuring the JVM

As part of configuring an application server, you might define settings that enhance the way your operating system uses of the Java virtual machine (JVM).

About this task

The Java virtual machine (JVM) is an interpretive computing engine responsible for running the byte codes in a compiled Java program. The JVM translates the Java byte codes into the native instructions of the host machine. The application server, being a Java process, requires a JVM in order to run, and to support the Java applications running on it. JVM settings are part of an application server configuration.

To view and change the JVM configuration for an application server's process, use the Java virtual machine page of the administrative console or use wsadmin to change the configuration through scripting.

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**. Then, under Server Infrastructure, click **Java and process management > Process definition**
2. Select **Control**.
3. Select **Java virtual machine**.
4. Specify values for the JVM settings as needed and click **OK**.
5. Click **Save** on the console task bar.
6. Restart the application server.

Example

“Configuring application servers for UCS Transformation Format” on page 288 provides an example that involves specifying a value for the **Generic JVM Arguments** property on the Java virtual machine page to enable UTF-8 encoding on an application server. Enabling UTF-8 allows multiple language encoding support to be used in the administrative console.

“Configuring JVM sendRedirect calls to use context root” on page 277 provides an example that involves defining a property for the JVM.

Java virtual machine settings

Use this page to view, and change the Java virtual machine (JVM) configuration settings of a process for an application server.

To view this administrative console page, connect to the administrative console and navigate to the Java virtual machine panel.

For the z/OS platform follow one of the following paths.

Application server	Click Servers > Server Types > WebSphere application servers > <i>server_name</i> . Then, in the Server Infrastructure section, click Java and process management > Process definition > Control > Java virtual machine
Deployment manager	Click System Administration > Deployment manager . Then, in the Server Infrastructure section, click Java and process management > Process definition > Control > Java virtual machine
Node agent	Click System Administration > Node agent > <i>node_agent</i> . Then, in the Server Infrastructure section, click Java and process management > Process definition > Java virtual machine

Classpath

Specifies the standard class path in which the Java virtual machine code looks for classes.

If you need to add a classpath to this field, enter each classpath entry into a separate table row. You do not have to add a colon or semicolon at the end of each entry.

The only classpaths that should be added to this field are the ones that specify the location of the following items:

- An inspection or monitoring tool to your system.
- JAR files for a product that runs on top of this product.
- JVM diagnostic patches or fixes.

Processing errors might occur if you add classpaths to this field that specify the location of the following items:

- JAR files for resource providers, such as DB2. The paths to these JAR files should be added to the relevant provider class paths.
- A user JAR file that is used by one or more of the applications that you are running on the product. The path to this type of JAR file should be specified within each application that requires that JAR file, or in server-associated shared libraries.
- An extension JAR file. If you need to add an extension JAR file to your system, you should use the `ws.ext.dirs` JVM custom property to specify the absolute path to this JAR file. You can also place the JAR file in the `WAS_HOME/lib/ext/` directory, but using the `ws.ext.dirs` JVM custom property is the recommended approach for specifying the path to an extension JAR file.

Data type

String

Boot classpath

Specifies bootstrap classes and resources for JVM code. This option is only available for JVM instructions that support bootstrap classes and resources.

If you need to add a classpath to this field, enter each classpath entry into a table row. You do not need to add the colon or semicolon at the end of each entry.

If you need to add multiple classpaths to this field, you can use either a colon (:) or semi-colon (;), depending on which operating system the node resides, to separate these classpaths.

The only classpaths that should be added to this field are the ones that specify the location of the following items:

- An inspection or monitoring tool to your system.
- JAR files for a product that runs on top of this product.
- JVM diagnostic patches or fixes.

Processing errors might occur if you add classpaths to this field that specify the location of the following items:

- JAR files for resource providers, such as DB2. The paths to these JAR files should be added to the relevant provider class paths.
- A user JAR file that is used by one or more of the applications that you are running on the product. The path to this type of JAR file should be specified within each application that requires that JAR file, or in server-associated shared libraries.
- An extension JAR file. If you need to add an extension JAR file to your system, you should use the `ws.ext.dirs` JVM custom property to specify the absolute path to this JAR file. You can also place the JAR file in the `WAS_HOME/lib/ext/` directory, but using the `ws.ext.dirs` JVM custom property is the recommended approach for specifying the path to an extension JAR file.

Verbose class loading

Specifies whether to use verbose debug output for class loading. The default is to not enable verbose class loading.

Data type	Boolean
Default	false

Verbose garbage collection

Specifies whether to use verbose debug output for garbage collection. The default is not to enable verbose garbage collection.

Data type	Boolean
Default	false

When this field is enabled, a report is written to the output stream each time the garbage collector runs. This report should give you an indication of how the Java garbage collection process is functioning.

You can check the `verboseGC` report to determine:

- How much time the JVM is spending performing garbage collection.
Ideally, you want the JVM to spend less than 5 percent of its processing time doing garbage collection. To determine the percentage of time the JVM spends in garbage collection, divide the time it took to complete the collection by the length of time since the last AF and multiply the result by 100. For example,
 $83.29/3724.32 * 100 = 2.236$ percent
- If the allocated heap is growing with each garbage collection occurrence.
To determine if the allocated heap is growing, look at the percentage of the heap that is remains unallocated after each garbage collection cycle, and verify that the percentage is not continuing to decline. If the percentage of free space continues to decline you are experiencing a gradual growth in the heap size from garbage collection to garbage collection. This situation might indicate that your application has a memory leak.

On the z/OS platform, you can also issue the MVS console command, `modify display, jvmheap`, to display JVM heap information. In addition, you can check the server activity and interval SMF records. The JVM heap size is also made available to PMI and can be monitored using the Tivoli Performance Viewer.

Verbose JNI

Specifies whether to use verbose debug output for native method invocation. The default is not to enable verbose Java Native Interface (JNI) activity.

Data type	Boolean
Default	false

Initial heap size

Specifies, in megabytes, the initial heap size available to the JVM code. If this field is left blank, the default value is used.

For z/OS, the default initial heap size for the controller is 48 MB, and the default initial heap size for the servant is 128 MB. These default values apply for both 31-bit and 64-bit configurations.

Note: These default values are sufficient for most applications.

Increasing this setting can improve startup. The number of garbage collection occurrences are reduced and a 10 percent gain in performance is achieved.

Increasing the size of the Java heap continues to improve throughput until the heap becomes too large to reside in physical memory. If the heap size exceeds the available physical memory, and paging occurs, there is a noticeable decrease in performance.

Maximum heap size

Specifies, in megabytes, the maximum heap size that is available to the JVM code. If this field is left blank, the default value is used.

For z/OS, the default maximum heap size is 256 MB. This default value applies for both 31-bit and 64-bit configurations.

Increasing the maximum heap size setting can improve startup. When you increase the maximum heap size, you reduce the number of garbage collection occurrences with a 10 percent gain in performance.

Increasing this setting usually improves throughput until the heap becomes too large to reside in physical memory. If the heap size exceeds the available physical memory, and paging occurs, there is a noticeable decrease in performance. Therefore, it is important that the value you specify for this property allows the heap to be contained within physical memory.

To prevent paging, specify a value for this property that allows a minimum of 256 MB of physical memory for each processor and 512 MB of physical memory for each application server. If processor utilization is low because of paging, increase the available memory, if possible, instead of increasing the maximum heap size. Increasing the maximum heap size might decrease performance rather than improving performance.

Note: These default values are appropriate for most applications. Enable the **Verbose garbage collection** property if you think garbage collection is occurring too frequently. If garbage collection is occurring too frequently, increase the maximum size of the JVM heap.

Debug mode

Specifies whether to run the JVM in debug mode. The default is to not enable debug mode support.

If you set the **Debug mode** property to true, then you must specify command-line debug arguments as values for the **Debug arguments** property.

Data type	Boolean
Default	false

Debug arguments

Specifies command-line debug arguments to pass to the JVM code that starts the application server process. You can specify arguments when the **Debug mode** property is set to true.

If you enable debugging on multiple application servers on the same node, verify that the same value is not specified for the address argument. The address argument defines the port that is used for debugging. If two servers, for which debugging is enabled, are configured to use the same debug port, the servers might fail to start properly. For example, both servers might still be configured with the debug argument address=7777, which is the default value for the debug address argument.

Data type	String
Units	Java command-line arguments

Generic JVM arguments

Specifies command-line arguments to pass to the Java virtual machine code that starts the application server process.

You can enter the following optional command-line arguments in the **Generic JVM arguments** field. If you enter more than one argument, enter a space between each argument.

Note: If the argument states that it is only for the IBM Developer Kit only, you cannot use that argument with the JVM from another provider, such as the Microsoft or Hewlett-Packard

- **hotRestartSync:**

Specify hotRestartSync if you want to enable the hot restart sync feature of the synchronization service. This feature indicates to the synchronization service that the installation is running in an environment where configuration updates are not made when the deployment manager is not active. Therefore, the service does not have to perform a complete repository comparison when the deployment manager or node agent servers restart. Enabling this feature improves the efficiency of the first synchronization operation after the deployment manager or a node agent restarts, especially for installations that include mixed release cells, use several nodes, and run several applications.

- **-Xquickstart**

Specify -Xquickstart if you want the initial compilation to occur at a lower optimization level than in default mode. Later, depending on sampling results, you can recompile to the level of the initial compile in default mode.

Note: Use -Xquickstart for applications where early moderate speed is more important than long run throughput. In some debug scenarios, test harnesses and short-running tools, you can improve startup time between 15-20 percent.

- **-Xverify:none**

Specify -Xverify:none if you want to skip the class verification stage during class loading. Using **-Xverify:none** disables Java class verification, which can provide a 10-15 percent improvement in startup time. However corrupted or invalid class data is not detected when this argument is specified. If corrupt class data is loaded, the JVM might behave in an unexpected manner, or the JVM might fail.

Note:

- Do not use this argument if you are making bytecode modifications, because the JVM might fail if any instrumentation error occurs.

- If you experience a JVM failure or the JVM behaves in an unexpected manner while this argument is in affect, remove this argument as your first step in debugging your JVM problem.

- **-Xnoclassgc**

Specify `-Xnoclassgc` if you want to disable class garbage collection. This argument results in more class reuse and slightly improved performance. However, the resources owned by these classes remain in use even when the classes are not being called. You can use the `verbose:gc` configuration setting if you want to monitor garbage collection. You can use the resulting output to determine the performance impact of reclaiming these resources. If the same set of classes are garbage collected repeatedly, you might want to disable class garbage collection. Class garbage collection is enabled by default.

- **-Xgcthreads**

Specify `-Xgcthreads` if you want to use several garbage collection threads at one time. This garbage collection techniques is known as *parallel garbage collection*. This argument is valid only for the IBM Developer Kit.

When entering this value in the **Generic JVM arguments** field, also enter the number of processors that are running on your machine. For example, if you have 3 processors running on your machine, enter `-Xgcthreads 3`. On a node with n processors, the default number of threads is n .

Note: You should use parallel garbage collection if your machine has more than one processor.

- **-Xnocompactgc**

Specify `-Xnocompactgc` if you want to disable heap compaction. Heap compaction is the most expensive garbage collection operation. If you are using the IBM Developer Kit, you should avoid heap compaction. If you disable heap compaction, you eliminate all associated overhead.

- **-Xgppolicy**

Specify `-Xgppolicy` to set the garbage collection policy. This argument is valid only for the IBM Developer Kit.

Set this argument to `optavgpause`, if you want concurrent marking used to track application threads starting from the stack before the heap becomes full. When this parameter is specified, the garbage collector pauses become uniform and long pauses are not apparent. However, using this policy reduces throughput because threads might have to do extra work.

Set this argument to `optthruput` if you want to optimize throughput and it does not create a problem if long garbage collection pauses occur. This is the default parameter, recommended setting.

- **-XX**

The Java Platform, Standard Edition 6 (Java SE 6) has generation garbage collection, which allows separate memory pools to contain objects with different ages. The garbage collection cycle collects the objects independently from one another depending on age. With additional parameters, you can set the size of the memory pools individually. To achieve better performance, set the size of the pool containing objects that have short life cycles, such that the objects in the pool are not kept through more than one garbage collection cycle. Use the `NewSize` and `MaxNewSize` parameters to specify the size of the new generation pool.

Objects that survive the first garbage collection cycle are transferred to another pool. Use the `SurvivorRatio` parameter to specify the size of the survivor pool. `SurvivorRatio`. You can use the object statistics that the Tivoli Performance Viewer collects, or include the `verbose:gc` argument in your configuration setting to monitor garbage collection statistics. If garbage collection becomes a bottleneck, specify the following arguments to customize the generation pool settings to better fit your environment.

```
-XX:NewSize=lower_bound
-XX:MaxNewSize=upper_bound
-XX:SurvivorRatio=new_ratio_size
```

Note: The default values for these arguments are: `NewSize=2m` `MaxNewSize=32m` `SurvivorRatio=2`. However, if you have a JVM that is configured with a heap size that is greater than 1 GB, use the values: `-XX:newSize=640m` `-XX:MaxNewSize=640m` `-XX:SurvivorRatio=16`, or set 50 to 60 percent of total heap size to a new generation pool.

- **-Xminf**

Specify `-Xminf` if you want to change the minimum free heap size percentage. The heap grows if the free space is below the specified amount. In reset enabled mode, this argument specifies the minimum percentage of free space for the middleware and transient heaps. The value specified for this argument is a floating point number, 0 through 1. The default is .3 (30 percent).

- **-server | -client**

Java HotSpot Technology in Java SE 6 uses an adaptive JVM containing algorithms that, over time, optimize how the byte code performs. The JVM runs in two modes, **-server** and **-client**. In most cases, use **-server** mode, which produces more efficient run-time performance over extended lengths of time.

If you use the default **-client** mode, the server startup time is quicker and a smaller memory footprint is created. However, this mode lowers extended performance. Use the **-server** mode, which improves performance, unless server startup time is of higher importance than performance. You can monitor the process size, and the server startup time to check the performance difference between using the **-client** and **-server** modes.

- **-Dcom.ibm.CORBA.RequestTimeout=*timeout_interval***

This argument only applies for z/OS. Specify the `-Dcom.ibm.CORBA.RequestTimeout=timeout_interval` argument to set the timeout period for responding to requests sent from the client. This argument uses the `-D` option. *timeout_interval* is the timeout period in seconds. If your network experiences extreme latency, specify a large value to prevent timeouts. If you specify a value that is too small, an application server that participates in workload management can time out before it receives a response.

Specify this argument only if your application is experiencing problems with timeouts. There are no recommended values for this argument.

- **-Dcom.ibm.websphere.wlm.unusable.interval=*interval***

This argument only applies for z/OS. Specify the `-Dcom.ibm.websphere.wlm.unusable.interval=interval` argument to change the value of the `com.ibm.websphere.wlm.unusable.interval` property when the workload management state of the client is refreshing too soon or too late. This property specifies, in seconds, the amount of time that the workload management client run time waits after it marks a server as unavailable before it attempts to contact the server again. This argument uses the `-D` option. The default value is 300 seconds. If the property is set to a large value, the server is marked as unavailable for a long period of time. This prevents the workload management refresh protocol from refreshing the workload management state of the client until after the time period has ended.

- **-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=**

This argument only applies for z/OS. Specify the `-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=` argument to indicate that storage for individual direct byte buffers should be released as soon as the buffer is no longer needed. The only supported value for this argument is `com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl`.

The direct byte buffers, that the JVM creates to handle request data, are allocated in the Language Environment (LE) heap instead of in the JVM heap. Typically, even if the direct byte buffers are no longer needed, the JVM does not release this native LE storage until the next garbage collection occurs. If the server is handling large requests, LE storage might become exhausted before the JVM runs a garbage collection cycle, causing the server to abnormally terminate (abend). Configuring the JVM with the following argument prevents these abends from occurring.

```
-Dcom.ibm.ws.buffermgmt.impl.WsByteBufferPoolManagerImpl=  
com.ibm.ws.buffermgmt.impl.ZOSWsByteBufferPoolManagerImpl
```

On the z/OS platform, you also need to specify this argument if you specify the `zaiFreeInitialBuffers` custom property for a TCP channel to have the channel release the initial read buffers used on new connections as soon as these buffers are no longer needed for the connection.

- **-Xshareclasses:none**

Specify the `-Xshareclasses:none` argument to disable the share classes option for a process. The share classes option, which is available with Java SE 6, lets you share classes in a cache. Sharing classes in a cache can improve startup time and reduce memory footprint. Processes, such as application servers, node agents, and deployment managers, can use the share classes option.

If you use this option, you should clear the cache when the process is not in use. To clear the cache, either call the `app_server_root/bin/clearClassCache.bat/sh` utility or stop the process and then restart the process.

Note:

- J2EE application classes running in an application server process are not added to the shared class cache.

Data type	String
Units	Java command-line arguments

Executable JAR file name

Specifies a full path name for an executable JAR file that the JVM code uses.

Data type	String
Units	Path name

Disable JIT

Specifies whether to disable the just-in-time (JIT) compiler option of the JVM code.

If you disable the JIT compiler, throughput decreases noticeably. Therefore, for performance reasons, keep JIT enabled.

Data type	Boolean
Default	false (JIT enabled)
Recommended	JIT enabled

Operating system name

Specifies JVM settings for a given operating system.

When the process starts, the process uses the JVM settings that are specified for the node as the JVM settings for the operating system.

Configuring JVM sendRedirect calls to use context root

If the `com.ibm.websphere.sendredirect.compatibility` property is not set and your application servlet code has statements such as `sendRedirect("/home.html")`, your Web browser might display messages such as *Error 404: No target servlet configured for uri: /home.html*.

About this task

Note: The `com.ibm.websphere.sendredirect.compatibility` property is deprecated. You should modify your applications to redirect non-relative URLs (those starting with a `/`) relative to the servlet container (`web_server_root`) instead of relative to the Web application context root.

To instruct the server to use the context root for that the application uses for `sendRedirect()` calls instead of using the document root for the Web server, configure the Java Virtual Machine (JVM) by setting the `com.ibm.websphere.sendredirect.compatibility` property to a `true` or `false` value.

1. Access the settings page for a property of the JVM.
 - a. In the administrative console, click **Servers > Server Types > Application servers**.
 - b. On the Application server page, click on the name of the server whose JVM settings you want to configure.
 - c. On the settings page for the selected application server, in the Server Infrastructure section, click **Java and process management > Process definition**.

- d. Select **Control**.
 - e. On the Process definition page, click **Java virtual machine**.
 - f. On the Java virtual machine page, click **Custom Properties**.
 - g. On the Custom properties page, click **New**.
2. On the settings page for a property, specify `com.ibm.websphere.sendredirect.compatibility` in the **Name** field, and either `true` or `false` in the **Value** field. Then click **OK**.
 3. Click **Save** on the console task bar.
 4. Stop the application server, and then restart the application server.

Java virtual machine custom properties

You can use the administrative console to change the values of Java virtual machine (JVM) custom properties.

To set custom properties, connect to the administrative console and navigate to the appropriate Java virtual machine custom properties page.

Application server	Click Servers > Server Types > WebSphere application servers > <i>server_name</i> , and then, in the Server Infrastructure section, click Java and process management > Process definition > Control > Java virtual machine > Custom properties
Deployment manager	Click System Administration > Deployment manager > Java and process management > Process definition > Control > Java virtual machine > Custom properties
Node agent	System Administration > Node agent > <i>nodeagent_name</i> > Java and process management > Process definition > Control > Java virtual machine > Custom properties

If the custom property is not present in the list of already defined custom properties, create a new property, and enter the property name in the Name field and a valid value in the Value field. Restart the server to complete your changes.

Note: Any custom property that begins with the string `was` is considered a system property. You can create a JVM custom property that starts with the string `was`, but you cannot use the administrative console to change the setting of such a custom property because any custom property that starts with the string `was` is not included in the list of available JVM custom properties that displays in the administrative console.

com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName

The EJB container should allow for the expansion of the CMP Connection Factor JNDI Name when a user's JNDI name contains a user defined Application Server variable. The custom property, `com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName`, makes it possible to expand the CMP Connection Factory JNDI Name.

If the value is **true**, which is the default, the EJB Container expands a variable when found in the CMP Connection Factory JNDI Name. If the value is set to **false**, the EJB Container does not expand a variable.

com.ibm.websphere.sib.webservices.useTypeSoapArray

You can pass messages directly to a bus destination by overriding the JAX-RPC client binding namespace and endpoint address. However:

- The default RPC-encoded Web services string array message that is generated might not interoperate successfully with some target service providers.
- The string array message produced is not exactly the same as the standard JAX-RPC equivalent, which can interoperate successfully.

Here are examples of the two different messages:

- Service integration bus message:

```
<partname env:encodingStyle='http://schemas.xmlsoap.org/soap/encoding/ xsi:type='ns1:ArrayOf_xsd_string'>
  <item xsi:type='xsd:anySimpleType'>namevalue</item>
</partname>
```

- JAX-RPC client message:

```
<partname xsi:type="soapenc:Array" soapenc:arrayType="xsd:string[1]">
  <item>namevalue</item>
</partname>
```

Set this property to `true` to modify the default behavior and send a string array message that is fully compatible with standard JAX-RPC. Setting this property modifies the default behavior for all outbound JMS Web services invocations sent from the service integration bus.

com.ibm.ws.sib.webservices.useSOAPJMSTextMessages

By default on WebSphere Application Server Version 6 or later, a SOAP over JMS Web service message sent by the Web services gateway is sent as a `JmsBytesMessage`, whereas on WebSphere Application Server Version 5.1 the Web services gateway sends a `JmsTextMessage`.

Set this property to `true` to modify the default behavior and send a compatible `JmsTextMessage`. Setting this property modifies the default behavior for all outbound JMS Web services invocations sent from the service integration bus.

com.ibm.websphere.ejbcontainer.expandCMPCFJNDIName

Use this Enterprise JavaBeans (EJB) custom property to expand the variables used in a container-managed persistence (CMP) connection factory Java™ Naming and Directory Interface (JNDI) name.

The EJB Container should allow for the expansion of the CMP connection factory JNDI name when a JNDI name contains a user-defined Application Server variable, although V6.1 does not support the expansion of variables. You need to use this property in order to expand the variables. You can enable or disable expansion.

To enable the expansion, the property value is `true`. To disable, use the value `false`.

The default is `true`.

If the value is `true`, the EJB container expands a variable found in the CMP connection factory JNDI name. If the value is `false`, the EJB container does not expand a variable.

com.ibm.websphere.application.updateapponcluster.waitforappsave

Specifies, in seconds, the amount of time that you want the deployment manager to wait for the extension tasks of the save operation to complete before starting the updated application.

Note: This property is only valid if it is specified for a deployment manager.

Usually during the save operation for an application update that is being performed using the rollout update process, the extension tasks of the save operation run as a background operation in a separate thread. If the main thread of the save operation completes before the synchronization portion of the rollout update process, the updated application fails to start properly.

When you add this custom property to your deployment manager settings, if the extension tasks of the save operation do not complete within the specified amount of time, the rollout update process stops the application update process, thereby preventing the application from becoming corrupted during the synchronization portion of the rollout update process.

The default value is 180.

com.ibm.ejs.sm.server.quiesceTimeout

Specifies, in seconds, the overall length of the quiesce timeout. If a request is still outstanding after this number of seconds, the server might start to shut down. For example, a value of 180 would be 3 minutes.

The default value is 180.

com.ibm.ejs.sm.server.quiesceInactiveRequestTime

Specifies, in milliseconds, how fast requests can come in and still be processed. For example, if you specify a value of 5000 for this property, the server does not attempt to shutdown until incoming requests are spaced at least 5 seconds apart. If the value specified for this property is too large, when the application server is stopped from the administrative console the following error message might be issued:

```
An error occurred while stopping Server1. Check the error logs for more information.
```

The default value is 5000 (5 seconds).

com.ibm.websphere.bean.delete.sleep.time

Use this property to specify the time between sweeps to check for timed out beans. The value is entered in seconds. For example, a value of 120 would be 2 minutes. This property also controls the interval in the Servant process that checks for timed out beans still visible to the enterprise bean container.

The default value is 4200 (70 minutes). The minimum value is 60 (1 minute). The value can be changed through the administrative console. To apply this property, you must specify the value in both the Control and Servant JVM Custom Properties.

com.ibm.websphere.deletejspclasses

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been deleted or updated. The default value for this property is `false`.

com.ibm.websphere.deletejspclasses.delete

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been deleted, but not after they have been updated. The default value for this property is `false`.

com.ibm.websphere.deletejspclasses.update

Use this property to indicate that you want to delete JavaServer Pages classes for all applications after those applications have been updated, but not after they have been deleted. The default value for this property is `false`.

com.ibm.websphere.management.application.fullupdate

Use this property to specify that when any of your applications are updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.

If this property is not specified, each changed file within an updated EAR file is individually updated and synchronized in the node. This process can be time consuming for large applications if a large number of files change.

Setting the `com.ibm.websphere.management.application.fullupdate` property to:

- `true` specifies that, when any of your applications are updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.

- `false` specifies that, when any of your applications are updated, you only want the changed files within that EAR file updated on the node and then synchronized.

Note: Use the `com.ibm.websphere.management.application.fullupdate.application_name` property if you only want to do a full replacement for a specific application instead of all of your applications.

com.ibm.websphere.management.application.fullupdate.application_name

Use this property to specify that when the specified application is updated, you want the binaries directory for that application erased and the content of the updated EAR file completely extracted.

If this property is not specified, each changed file within the updated EAR file for the specified application is individually updated and synchronized in the node. This process can be time consuming for large applications if a large number of files change.

Setting the `com.ibm.websphere.management.application.fullupdate.application_name` property to:

- `true` specifies that when the specified application is updated, you want the binaries directory erased and the content of the updated EAR file completely extracted.
- `false` that when the specified application is updated, you only want the changed files updated on the node and then synchronized.

Note: Use the `com.ibm.websphere.management.application.fullupdate` property if you want the binaries directory erased and the content of the updated EAR file completely extracted whenever any of your applications are updated.

com.ibm.websphere.management.application.sync.recycleappsv5

Use this property to specify that you want your application recycling behavior to work the same way as this behavior worked in Version 5.x of the product.

In Version 6.x and higher, after an application update or edit operation occurs, depending on which files are modified, either the application or its modules are automatically recycled. This recycling process occurs for all application configuration file changes, and all non-static file changes.

However, in Version 5.x of the product, an application is recycled only if the Enterprise Archive (EAR) file itself is updated, or if the binaries URL attribute changes. An application is not recycled if there is a change to the application configuration file.

Setting the `com.ibm.websphere.management.application.sync.recycleappsv5` property to:

- `true` specifies that you want your application recycling behavior to work the same way as this behavior worked in Version 5.x of the product.
- `false` specifies that you want your application recycling behavior to work according to the Version 6.x and higher behavior schema.

The default value for this custom property is `false`.

Note: You must define this property in the node agent JVM. However, when defining this property, you can specify a scope of cell if you want the setting to apply to all of the nodes within a specific cell. If this property is set at both the cell and node agent level, the node agent setting takes precedence for that particular node agent.

com.ibm.websphere.management.jmx.random

Use this property to enable the controller to randomly select an initial servant from the servant pool to process a Java Management Extensions (JMX) connector requests instead of automatically assigning the request to the hot servant.

By default, when multiple servants are enabled and the application server receives a JMX connection request, the application server assigns the request to the first servant, which is also referred to as the hot servant. This strategy minimizes the risk that the request is assigned to a servant that is paged out. However, if the first servant has a heavy workload, requests to that servant eventually fail. Therefore, the advantage of using the random algorithm is that the assigned servant is probably not already handling a lot of other requests. The disadvantage of using the random algorithm is that the selected servant might be paged out and have to be paged back in before it can handle the request.

Setting the `com.ibm.websphere.management.jmx.random` property to:

- `true` specifies that the controller will randomly select an initial servant from the servant pool to process a JMX connector requests.
- `false` specifies that the controller will assign all JMX connector requests to the hot servant.

com.ibm.websphere.management.registerServerIORWithLSD

Use this property to control whether a federated server registers with the Location Service Daemon (LSD). Normally, a federated server requires the node agent to be running. To direct the server to not register with the LSD and remove its dependency on an active node agent, the `com.ibm.websphere.management.registerServerIORWithLSD` JVM custom property must be set to `false`, and the `ORB_LISTENER_ADDRESS` configuring inbound transports must be set to a value greater than 0 so that the ORB listens at a fixed port. The setting for this property is ignored if the `ORB_LISTENER_ADDRESS` property is set to 0 (zero) or is not specified, and the federated server registers with the LSD.

Set this property to `false` if you want the server to run even when the node agent is not running. When this property is set to `true`, the federated server registers with the LSD. The default value for this custom property is `true`.

When you set the `com.ibm.websphere.management.registerServerIORWithLSD` property to `false`, the server does not notify the node agent when it dynamically assigns the `ORB_LISTENER_ADDRESS` port. There also will not be any indirect Interoperable Object References (IORs) that the node agent can resolve to a server. All of the IORs become direct, which means that the node agent can only contact that server if a static `ORB_LISTENER_ADDRESS` has been assigned to that server.

Note: If you set the `com.ibm.websphere.management.registerServerIORWithLSD` property to `false`, you should also create a static routing table to enable static routing. Enabling static routing ensures that workload management (WLM) continues to function properly.

com.ibm.websphere.network.useMultiHome

In a multihomed environment where the product is restricted to listen only on a specific IP address for Discovery and SOAP messages, set this property to `false` for the deployment manager, all application servers and all node agents. By default, the value of the property is `true` and the application server listens on all IP addresses on the host for Discovery and SOAP messages. If the property is set to `false`, the product only listens for Discovery and SOAP messages on the configured host name. If you set the property to `false`, you must also:

- Have a host name configured on the product that resolves to a specific IP address.
- Ensure that the end point property for the deployment manager, all application servers, and all node agents is set to this host name. For the deployment manager, the end points that must be set are `CELL_DISCOVERY_ADDRESS` and `SOAP_CONNECTOR_ADDRESS`. For the node agent and application servers, only the `SOAP_CONNECTOR_ADDRESS` end point must be set.

You can change the value through the administrative console. Modify the defaults by setting the value for the server, deployment manager, and node agent. In order for these changes to take place, you must restart the server.

If you cannot contact the server, check the setting for `com.ibm.websphere.network.useMultihome` to ensure it is correct. You can change the value through the administrative console. Modify the defaults by setting the value for the server, deployment manager, and node agent. You must restart the server before these changes take effect.

`com.ibm.websphere.webservices.attachments.maxMemCacheSize`

Use this property to specify, in kilobytes, the maximum size of a Web services attachment that can be written to memory. For example, if your Web service needs to send 20 MB attachments, set the property to 20480.

When determining a value for this property, remember that the larger the maximum cache size, the more impact there is on performance, and, potentially, to the Java heap.

If you do not specify a value for this property, the maximum memory that is used to cache attachments is 32 KB, which is the default value for this property.

`com.ibm.ws.pm.checkingDBconnection`

Use this property to specify whether the persistence manager is to continue checking the availability of a database, that was previously marked as unavailable, until a connection with that database is successfully established.

If a database service is down when the persistent manager attempts to establish a connection to that database, the database is marked as unavailable. Typically, the persistent manager does not re-attempt to establish a connection after a database is marked as unavailable. If you set this property to `true`, the persistence manager continues to check the availability of the database until it is able to successfully establish a connection to that database.

The default value for this property is `false`.

`com.ibm.ws.webservices.contentTransferEncoding`

Use this property to specify a range of bits for which XML-encoding is disabled. Typically any integer that is greater than 127 is XML-encoded. When you specify this property:

- Web services disables encoding for integers that fall within the specified range.
- The HTTP transport message contains a `ContentTransferEncoding` header that is set to the value that is specified for this custom property.

Specify `7bit`, if you only want integers greater than 127 encoded. Specify `8bit`, if you only want integers greater than 255 encoded. Specify `binary`, if you want encoding disabled for all integers.

The default value is `7bit`.

`com.ibm.ws.webservices.ignoreUnknownElements`

Use this property to control whether clients can ignore extra XML elements that are sometimes found within literal SOAP operation responses.

Setting this property to `true` provides you with the flexibility of being able to update your server code to include additional response information, without having to immediately update your client code to process this additional information. However, when this functionality is enabled, the checking of SOAP message against the expected message structure is more relaxed than when this property is set to `false`.

`com.ibm.ws.webservices.suppressHTTPRequestPortSuffix`

Use this property to control whether a port number can be left in an HTTP POST request that sends a SOAP message.

Some Web service implementations do not properly tolerate the presence of a port number within the HTTP POST request that sends the SOAP message. If you have a Web service client that needs to

inter-operate with Web service that cannot tolerate a port number within an HTTP POST request that sends a SOAP message, set this custom property to `true`.

When you set this property to `true`, the port number is removed from the HTTP POST request before it is sent.

Note: You must restart the server before this configuration setting takes affect.

The default value for this custom property is `false`.

com.ibm.websphere.ejb.UseEJB61FEPScanPolicy

Use this property to control whether the product scans pre-Java EE 5 modules for additional metadata during the application installation process or during server startup. By default, these legacy EJB modules are not scanned.

The default value for this custom property is `false`.

You must set this property to `true` for each server and administrative server that requires a change in the default value.

com.ibm.websphere.webservices.UseWSFEP61ScanPolicy

Use this property to control whether the product scans WAR 2.4 and earlier modules for JAXWS components and semi-managed service clients. By default, these legacy WAR modules are only scans for semi-managed service clients.

The default value for this custom property is `false`.

You must set this property to `true` for each server and administrative server that requires a change in the default value.

com.ibm.ws.ws.wsba.protocolmessages.twoway

Use this property to improve the performance of an application server that is handling requests for Web Services Business Activities (WS-BA). Specifying `true` for this custom property improves application server performance when WS-BA protocol messages are sent between two application servers. The default value for this property is `true`.

Note: If you decide to use this custom property, the property must be set on the application server that initiates the requests. It does not have to be set on the application server that receives the requests.

invocationCacheSize

Use this property to control the size of the invocation cache. The invocation cache holds information for mapping request URLs to servlet resources. A cache of the requested size is created for each worker thread that is available to process a request. The default size of the invocation cache is 50. If more than 50 unique URLs are actively being used (each JavaServer Page is a unique URL), you should increase the size of the invocation cache.

A larger cache uses more of the Java heap, so you might also need to increase the maximum Java heap size. For example, if each cache entry requires 2KB, maximum thread size is set to 25, and the URL invocation cache size is 100; then 5MB of Java heap are required.

You can specify any number higher than 0 for the cache size. Setting the value to zero disables the invocation cache.

ODCClearMessageAge

Use this property to establish a length of time, specified in milliseconds, after which an ODC message is removed from the bulletin board, even if the receiver has not acknowledged the message. Specifying a value for this property helps prevent the build up of messages that, for some reason, do not get acknowledged.

You can specify any positive integer as a value for this property, but a value of 300000 (5 minutes) or higher is recommended to avoid premature removal of messages.

The default value is 300000 milliseconds.

was.xcfmonitor.enabled

Use this property to enable the use of the z/OS Cross-system Coupling Facility (XCF) for monitoring application server status. When this property is set to true, node agents can use XCF to monitor their application servers, application servers can use XCF to monitor their node agents, and an administrative agent server can use XCF to monitor its registered application servers. When this property is set to false, the TCP/IP pinging method is used to monitor application server and node agent status.

The default value is true.

Preparing to host applications

Rather than use the default application server provided with the product, you can configure a new server and set of resources.

About this task

The default application server and a set of default resources are available to help you begin quickly. You can choose instead to configure a new server and set of resources. Here is what you need to do in order to set up a runtime environment to support applications.

1. Configure an application server.
2. Create a virtual host.
3. Configure a Web container. See the *Administering applications and their environment* PDF for more information.
4. Configure an EJB container. See the *Administering applications and their environment* PDF for more information.
5. Create resources for data access. See the *Administering applications and their environment* PDF for more information.
6. Create a JDBC provider and data source. See the *Administering applications and their environment* PDF for more information.
7. Create a URL and URL provider. See the *Administering applications and their environment* PDF for more information.
8. Create a mail session. See the *Administering applications and their environment* PDF for more information.
9. Create resources for session support. See the *Administering applications and their environment* PDF for more information.
10. Configure a Session Manager. See the *Administering applications and their environment* PDF for more information.

Configuring multiple network interface support

Application servers, by default, are configured to use all of the network interfaces that are available for them to use. You can change this configuration such that an application server only uses a specific network interface. However, you cannot configure it to use a subgroup of interfaces. For example, if you have three ethernet adapters, you cannot configure an application server to use two of the three adapters.

About this task

When an application server is configured to use all network interfaces, if it opens a socket on port 9901 on a machine with two TCP/IP addresses, it opens port 9901 on both IP addresses.

When an application server is configured to use a specific network interface, it only communicates on that one network interface. For example, on a Windows operating system, if an application server opens a socket on port 7842 on an ethernet adapter with an address of 192.168.1.150, the netstat output displays 192.168.1.150.7842 in the Local Address field, indicating that port 7842 is only bound to 192.168.1.150.

If you have more than one network interface and you want to use each one separately, you must have a separate configuration profile for each interface. When network interfaces are used separately, a separate node agent is required for each network interface that has an application server running on it. Two application servers bound to two separate network interfaces on the same machine cannot be in the same node because they have different TCP/IP addresses.

Note:

- If you want a specific application server to use a single network interface, perform the following steps for that application server.
 - If you want an entire node to use a single network interface, perform the following steps for your node agent and all the application servers in that node.
 - If you want an entire cell to use a single network interface, perform the following steps for the deployment manager, node agent, and all the application servers in the node.
 - When performing the following steps, do not specify localhost, a loop back address, such as 127.0.0.1, or an * (asterisk) for the TCP/IP addresses.
1. Update the com.ibm.CORBA.LocalHost and com.ibm.ws.orb.transport.useMultiHome Object Request Broker (ORB) custom properties.
 - a. In the administrative console, navigate to the indicated page.
 - For an application server, click **Servers > Server Types > WebSphere application servers > *server_name* > Container Settings > Container services > ORB Service**. Then in the Additional Properties section, click **Custom properties**.
 - For a deployment manager, click **System Administration > Deployment manager**. In the Additional Properties section, click **ORB Service**. Then, under Additional properties on the **ORB Service** page, click **Custom properties**.
 - For a node agent, click **System Administration > Node agents *node_agent***. In the Additional Properties section, click **ORB Service**. Then, under Additional properties on the **ORB Service** page, click **Custom properties**.
 - b. Select the com.ibm.CORBA.LocalHost custom property and specify an IP address or hostname in the Value field. Do not set this property to either localhost or *.
If the com.ibm.CORBA.LocalHost property is not in the list of already defined custom properties, click **New** and then enter com.ibm.CORBA.LocalHost in the Name field and specify an IP address or hostname in the Value field.
 - c. Select the com.ibm.ws.orb.transport.useMultiHome custom property and specify false in the Value field. If the com.ibm.ws.orb.transport.useMultiHome property is not in the list of already defined custom properties, click **New** and then enter com.ibm.ws.orb.transport.useMultiHome in the Name field and specify false in the Value field.

2. Update the `daemon_protocol_iiop_listenIPAddress` WebSphere variable to indicate the IP addresses to which you want the location service daemon to bind.
 - a. In the administrative console, click **Environment > WebSphere variables**.
 - b. Select the `DAEMON_protocol_iiop_listenIPAddress` variable and specify * to specify bind all, or specify a specific IP address in the Value field. If the `DAEMON_protocol_iiop_listenIPAddress` variable is not in the list of already defined variables, click **New** and then enter `DAEMON_protocol_iiop_listenIPAddress` in the Name field and specify the appropriate value in the Value field.
3. Update the Java virtual machine (JVM) `com.ibm.websphere.network.useMultiHome` custom property for discovery and SOAP connections.
 - a. In the administrative console, navigate to the indicated page.
 - For an application server, click **Servers > Server Types > WebSphere application servers > server_name > Java process management > Process definition > process_type > Java virtual machine > Custom properties**
 - For a deployment manager, click **System Administration > Deployment manager > Java process management > Process definition > process_type > Java virtual machine > Custom properties**.
 - For a node agent, click **System Administration > Node agents > node_agent > Java process management > Process definition > Control > Java virtual machine > Custom properties**.
 - b. Select the `com.ibm.websphere.network.useMultiHome` custom property and specify `false` in the Value field. If the `com.ibm.websphere.network.useMultiHome` property is not in the list of already defined custom properties, click **New** and then enter `com.ibm.websphere.network.useMultiHome` in the Name field and specify `false` in the Value field.
4. Update the host name for TCP/IP connections.
 - a. In the administrative console, navigate to the indicated page.
 - For an application server, click **Servers > Server Types > WebSphere application servers > server_name**, and then, in the Additional Properties section, click **Ports**.
 - For a deployment manager, click **System Administration > Deployment manager**, and then, in the Additional Properties section, click **Ports**.
 - For a node agent, click **System Administration > Node agents > node_agent**, and then, in the Additional Properties section, click **Ports**.
 - b. Update the Host field for each of the listed ports to the value specified for the `com.ibm.CORBA.LocalHost ORB` custom property in the first step. When you finish, none of the entries listed in the Host column should contain an * (asterisk).
5. Change the Initial State setting for each of the JMS servers to Stopped .
 - a. In the administrative console, click **Servers > Server Types > JMS servers >**
 - b. Click one of the listed JMS servers and change the value specified for the Initial State field to Stopped.
 - c. Repeat the previous step until the Initial State setting for all of the listed JMS servers is Stopped.
6. Change the Initial State setting for each of the listener ports to Stopped .
 - a. In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**.
 - b. Under Communications, click **Messaging > Message Listener Service > Listener Ports**.
 - c. Click one of the listed listener ports and change the value specified for the Initial State field to Stopped.
 - d. Repeat the previous step until the Initial State setting for all of the listed listener ports is Stopped.
7. Save your changes.
 - a. In the administrative console, click **System administration > Save Changes to Master Repository**.

- b. Select Synchronize changes with nodes, and then click **Save**.
8. Stop and restart all the affected servers, node agents, and the deployment manager.

Results

You have configured an installation of WebSphere Application Server to communicate on one, and only one network interface on a machine that has more than one network interface.

Example

This example creates two nodes, each using a separate network interface, on a machine that has at least two network interfaces:

1. Use the Profile Management tool to create an application server and federate it into the desired cell.
2. Use the Profile Management tool to create an application server profile, specifying a host name that is different than the host name used for the previously created application server. Federate this application server into the desired cell.
3. Start the node agent and application server that are configured to the first network interface. Follow the preceding steps for the node agent and application server to prepare this node to communicate on the network interface you specified when you configured this application server.
4. Start the second node agent and application server. Follow the preceding steps for the node agent and application server to prepare this node to communicate only on the network interface that you specified when you configured the second application server.
5. Stop all of the node agents and application servers that you created in this example.
6. Restart all of these node agents and application servers.

You have two separate nodes running on two different network interfaces.

What to do next

If you are using a standalone Java client or server to communicate with WebSphere Application Server, and you are using the WebSphere Application Server Software Development Kit (SDK), add the following properties to your Java command to enable the ORB for your application to communicate with a specific network interface.

```
-Dcom.ibm.ws.orb.transport.useMultiHome=false  
-Dcom.ibm.CORBA.LocalHost=host_name
```

host_name is the TCP/IP address or *hostname* of the network interface for the ORB to use.

Note: Do not set *host_name* to localhost, a loop back address, such as 127.0.0.1, or an * (asterisk).

Configuring application servers for UCS Transformation Format

You can use the `client.encoding.override=UTF-8` JVM argument to configure an application server for UCS Transformation Format. This format enables an application server to handle most character encodings, including specialized mathematical and technical symbols.

About this task

The `client.encoding.override=UTF-8` argument is provided for backwards compatibility. You should only specify this argument if you require multiple language encoding support in the administrative console and there is no other way for you to set the request character encoding required to parse post and query strings.

Before configuring an application server for UCS Transformation Format, you should try to either:

- Explicitly set the ServletRequest Encoding inside of the JSP or Servlet that is receiving the POST and or query string data, which is the preferred J2EE solution, or
- Enable the autoRequestEncoding, option, which uses the client's browser settings to determine the appropriate character encoding. Older browsers might not support this option.

Note: If the client.encoding.override=UTF-8 JVM argument is specified, the autoRequestEncoding option does not work even if it is enabled. Therefore, when an application server receives a client request, it checks to see if the charset option is set on the content type header of the request:

1. If it is set, the application server uses the content type header for character encoding.
2. If it is not set, the application server uses the character encoding that is specified for the default.client.encoding system property.
3. If neither charset nor the default.client.encoding system property is set, the application server uses the ISO-8859-1 character set.

The application server never checks for an Accept-Language header. However, if the autoRequestEncoding option is working, the application server checks for an Accept-Language header before checking to see if a character encoding is specified for the default.client.encoding system property.

To configure an application server for UCS Transformation Format:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers**, and select the server that you want to enable for UCS Transformation Format.
2. Then, in the Server Infrastructure section, click **Java and process management > Process definition > Control > Java virtual machine**.
3. Specify `-Dclient.encoding.override=UTF-8` for the **Generic JVM Arguments** property, and click **OK**. When this argument is specified, UCS Transformation Format is used instead of the character encoding that would be used if the autoRequestEncoding option was in effect.
4. Click **Save** to save your changes.
5. Restart the application server.

Results

The application server uses UCS Transformation Format for encoding.

Tuning application servers

The product contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application.

About this task

The following steps describe various tuning tasks that may improve your application server performance. You can choose to implement any of these application server settings. These steps can be performed in any order.

1. **Tune the object request broker.** An Object Request Broker (ORB) manages the interaction between clients and servers, using the Internet InterORB Protocol (IIOP). It supports client requests and responses received from servers in a network-distributed environment. You can use the following parameters to tune the ORB:
 - Set **Pass by reference (com.ibm.CORBA.iiop.noLocalCopies)** as described in the *Tuning guide* PDF.
 - Set the **com.ibm.CORBA.FragmentSize** as described in the *Administering applications and their environment* PDF.
2. **Tune the XML parser definitions.**

- **Description:** Facilitates server startup by adding XML parser definitions to the `jaxp.properties` and `xerces.properties` files in the `${app_server_root}/jre/lib` directory. The `XMLParserConfiguration` value might change as new versions of Xerces are provided.

- **How to view or set:** Insert the following lines in both files:

```
javax.xml.parsers.SAXParserFactory=org.apache.xerces.jaxp.SAXParserFactoryImpl
javax.xml.parsers.DocumentBuilderFactory=org.apache.xerces.jaxp.
    DocumentBuilderFactoryImpl
org.apache.xerces.xni.parser.XMLParserConfiguration=org.apache.xerces.parsers.
    StandardParserConfiguration
```

- **Default value:** None
- **Recommended value:** None

3. Tune the dynamic cache service.

Using the dynamic cache service can improve performance. See the *Administering applications and their environment* PDF for information about using the dynamic cache service and how it can affect your application server performance.

4. Tune the EJB container.

An Enterprise JavaBeans (EJB) container is automatically created when you create an application server. After the EJB container is deployed, you can use the following parameters to make adjustments that improve performance.

- Set the **Cleanup interval** and the **Cache size** as described in the *Administering applications and their environment* PDF.
- **Break CMP enterprise beans into several enterprise bean modules** while assembling EJB modules.

See also the *Tuning guide* PDF.

5. Tune the session management.

The installed default settings for session management are optimal for performance. See the *Tuning guide* PDF for more information about tuning session management.

6. Tune the data sources and associated connection pools.

A data source is used to access data from the database; it is associated with a pool of connections to that database.

7. Tune the URL invocation cache.

Each JavaServer Page is a unique URL. If you have more than 50 unique URLs that are actively being used, increase the value specified for the `invocationCacheSize` JVM custom property. This property controls the size of the URL invocation cache.

Each JavaServer Page is a unique URL. If you have more than 50 unique URLs that are actively being used, increase the value specified for the `invocationCacheSize` JVM custom property. This property controls the size of the URL invocation cache. See the *Administering applications and their environment* PDF for more information on how to change this property.

8. Change how frequently the recovery log service attempts to compress any logstreams that application components are using.

The Transaction Service `RLS_LOGSTREAM_COMPRESS_INTERVAL` custom property can be set to a value larger than the default value if the Transaction Service is the only application component using a logstream. If none of your components are configured to use a logstream, you can set this property to 0 (zero) to disable this function.

Web services client to Web container optimized communication

To improve performance, there is an optimized communication path between a Web services client application and a Web container that are located in the same application server process. Requests from the Web services client that are normally sent to the Web container using a network connection are delivered directly to the Web container using an optimized local path. The local path is available because the Web services client application and the Web container are running in the same process.

This direct communication eliminates the need for clients and web containers that are in the same process to communicate over the network. For example, a Web services client might be running in an application

server. Instead of accessing the network to communicate with the Web container, the Web services client can communicate with the Web container using the optimized local path. This optimized local path improves the performance of the application server by enabling Web services clients and Web containers to communicate without using network transports.

In a clustered environment, there is typically an HTTP server (such as IBM HTTP server) that handles incoming client requests, distributing them to the correct application server in the cluster. The HTTP server uses information about the requested application and the defined virtual hosts to determine which application server receives the request. The Web services client also uses the defined virtual host information to determine whether the request can be served by the local Web container. You must define unique values for the host and port on each application server. You cannot define the values of host and port as wild cards denoted by the asterisk symbol (*) when you enable the optimized communication between the Web services application and the Web container. Using wild cards indicate that the local Web container can handle Web services requests for all destinations.

The optimized local communication path is disabled by default. You can enable the local communication path with the `enableInProcessConnections` custom property. Before configuring this custom property, make sure that you are not using wild cards for host names in your Web container end points. Set this property to **true** in the Web container to enable the optimized local communication path. When disabled, the Web services client and the Web container communicate using network transports.

For information about how to configure the `enableInProcessConnections` custom property, see the *Administering applications and their environment* PDF.

When the optimized local communication path is enabled, logging of requests through the local path uses the same log attributes as the network channel chain for the Web container. To use a different log file for in process requests than the log file for network requests, use a custom property on the HTTP Inbound Channel in the transport chain. Use the `localLogFilenamePrefix` custom property to specify a string that is added to the beginning of the network log file name to create a file name that is unique. Requests through the local process path are logged to this specified file. For example, if the log filename is `../httpaccess.log` for a network chain, and the `localLogFilenamePrefix` custom property is set to "local" on the HTTP channel in that transport chain, the local log file name for requests to the host associated with that chain is `/localhttpaccess.log`.

Note: If you specify a value for the `localLogFilenamePrefix` custom property, you must also set the `accessLogFileName` HTTP channel custom property to the fully qualified name of the log file you want to use for in process requests. You cannot specify a variable, such as `$(SERVER_LOG_ROOT)`, as the value for this custom property.

Chapter 5. Balancing workloads with clusters

You should use server clusters and cluster members to monitor and manage the workloads of application servers.

Before you begin

You should understand your options for configuring application servers. To assist you in understanding how to configure and use clusters for workload management, consider this scenario. Client requests are distributed among the cluster members on a single machine. A *client* refers to any servlet, Java application, or other program or component that connects the end user and the application server that is being accessed.

In more complex workload management scenarios, you can distribute cluster members within the same sysplex.

About this task

Perform the following steps if you decide to use clusters to balance your workload.

1. Decide which application server you want to cluster.
2. Decide whether you want to replicate data. Replication is a service that transfers data, objects, or events among application servers.

You can create a replication domain when creating a cluster.

3. Deploy the application onto the application server.
4. Create a cluster.

After configuring the application server and the application components exactly as you want them to be, create a cluster. The original server instance becomes a cluster member that is administered through the cluster.

5. Create one or more cluster members.
6. Start the cluster.

When you start the cluster, all of the application servers that are members of that cluster start. Workload management automatically begins after the cluster members start.

7. After the cluster is running, you can perform the following tasks:
 - Stop the cluster.
 - Upgrade the applications that are installed on the cluster members.
 - Detect and handle problems with server clusters and their workloads.

Clusters and workload management

Clusters are sets of servers that are managed together and participate in workload management. Clusters enable enterprise applications to scale beyond the amount of throughput capable of being achieved with a single application server. Clusters also enable enterprise applications to be highly available because requests are automatically routed to the running servers in the event of a failure. The servers that are members of a cluster can be on different host machines. In contrast, servers that are part of the same node must be located on the same host machine. A cell can include no clusters, one cluster, or multiple clusters.

Servers that belong to a cluster are *members* of that cluster set and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data. One cluster member might be running on a huge multi-processor enterprise server system, while another member of that same cluster might be running on

a smaller system. The server configuration settings for each of these two cluster members are very different, except in the area of application components assigned to them. In that area of configuration, they are identical. This allows client work to be distributed across all the members of a cluster instead of all workload being handled by a single application server.

When you create a cluster, you make copies of an existing application server template. The template is most likely an application server that you have previously configured. You are offered the option of making that server a member of the cluster. However, it is recommended that you keep the server available only as a template, because the only way to remove a cluster member is to delete the application server. When you delete a cluster, you also delete any application servers that were members of that cluster. There is no way to preserve any member of a cluster. Keeping the original template intact allows you to reuse the template if you need to rebuild the configuration.

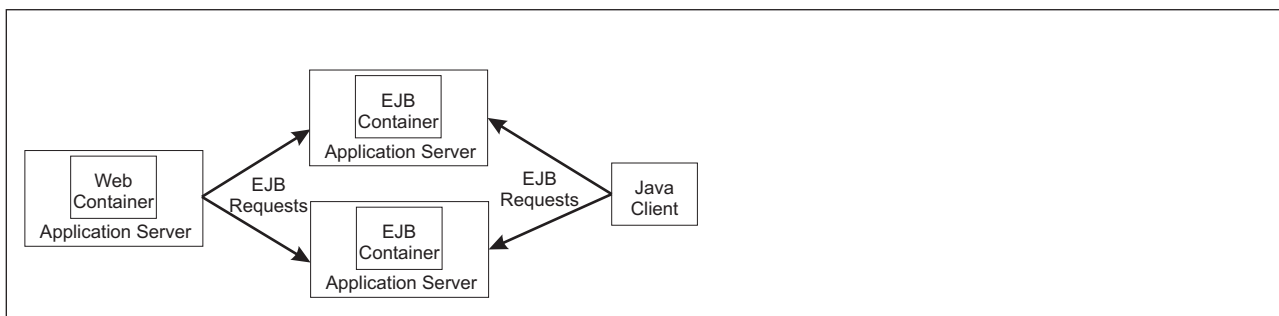
A *vertical cluster* has cluster members on the same node, or physical machine. A *horizontal cluster* has cluster members on multiple nodes across many machines in a cell. You can configure either type of cluster, or have a combination of vertical and horizontal clusters.

You can use the administrative console to specify a weight for a cluster member. The weight you assign to a cluster member should be based on its approximate, proportional ability to do work. The weight value specified for a specific member is only meaningful in the context of the weights you specify for the other members within a cluster. The weight values do not indicate absolute capability. If a cluster member is unavailable, the Web server plug-in temporarily routes requests around that cluster member.

For example, if you have a cluster that consists of two members, assigning weights of 1 and 2 causes the first member to get approximately 1/3 of the workload and the second member to get approximately 2/3 of the workload. However, if you add a third member to the cluster, and assign the new member a weight of 1, approximately 1/4 of the workload now goes to the first member, approximately 1/2 of the workload goes to the second member, and approximately 1/4 of the workload goes to the third member. If the first cluster member becomes unavailable, the second member gets approximately 2/3 of the workload and third member gets approximately 1/3 of the workload.

The weight values only approximate your load balance objectives. There are other application dependencies, such as thread concurrency, local setting preferences, affinity, and resource availability that are also factors in determining where a specific request is sent. Therefore, do not use the exact pattern of requests to determine the weight assignment for specific cluster members.

Workload management for EJB containers can be performed by configuring the Web container and EJB containers on separate application servers. Multiple application servers can be clustered with the EJB containers, enabling the distribution of enterprise bean requests between EJB containers on different application servers.



In this configuration, EJB client requests are routed to available EJB containers in a round robin fashion based on assigned server weights. The EJB clients can be servlets operating within a Web container, stand-alone Java programs using RMI/IIOP, or other EJBs.

The server weighted round robin routing policy ensures a balanced routing distribution based on the set of server weights that have been assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers. The policy ensures the desired distribution, based on the weights assigned to the cluster members.

You can set up workload management to balance the tasks between different clusters.

You can choose to have requests sent to the node on which the client resides as the preferred routing. In this case, only cluster members on that node are chosen (using the round robin weight method). Cluster members on remote nodes are chosen only if a local server is not available.

Multiple servers that can service the same client request form the basis for failover support. If a server fails while processing a client request, the failed request can be rerouted to any of the remaining cluster members. Even if several servers fail, as long as at least one cluster member is running, client requests continue to be serviced.

Techniques for managing state

Multiple machine scaling techniques rely on using multiple copies of an application server; multiple consecutive requests from various clients can be serviced by different servers. If each client request is completely independent of every other client request, it does not matter if consecutive requests are processed on the same server. However, in practice, client requests are not independent. A client often makes a request, waits for the result, then makes one or more subsequent requests that depend on the results received from the earlier requests.

This sequence of operations on behalf of a client falls into two categories:

Stateless

A server processes requests based solely on information provided with each request and does not rely on information from earlier requests. The server does not need to maintain state information between requests.

Stateful

A server processes requests based on both the information provided with each request and information stored from earlier requests. The server needs to access and maintain state information generated during the processing of an earlier request.

For stateless interactions, it does not matter whether different requests are processed by different servers. However, for stateful interactions, the server that processes a request needs access to the state information necessary to service that request. Either the same server can process all requests that are associated with the same state information, or the state information can be shared by all servers that require it. In the latter case, accessing the shared state information from the same server minimizes the processing overhead associated with accessing the shared state information from multiple servers.

The load distribution facilities in the product use several different techniques for maintaining state information between client requests:

- Session affinity, where the load distribution facility recognizes the existence of a client session and attempts to direct all requests within that session to the same server.
- Transaction affinity, where the load distribution facility recognizes the existence of a transaction and attempts to direct all requests within the scope of that transaction to the same server.
- Server affinity, where the load distribution facility recognizes that although multiple servers might be acceptable for a given client request, a particular server is best suited for processing that request.

Workload management (WLM) for z/OS

Workload management optimizes the distribution of incoming work requests to the application servers, enterprise beans, servlets, and other objects that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

For details on workload management, see *z/OS MVS Planning: Workload Management*, which is available on the z/OS Internet Library Web site. You might also find *z/OS MVS Programming: Workload Management Services* helpful.

When you are using workload management on z/OS, you can define workload management policies for your application servers. To get started, you do not need to define special classification rules and work qualifiers, but you might want to define them for your production system.

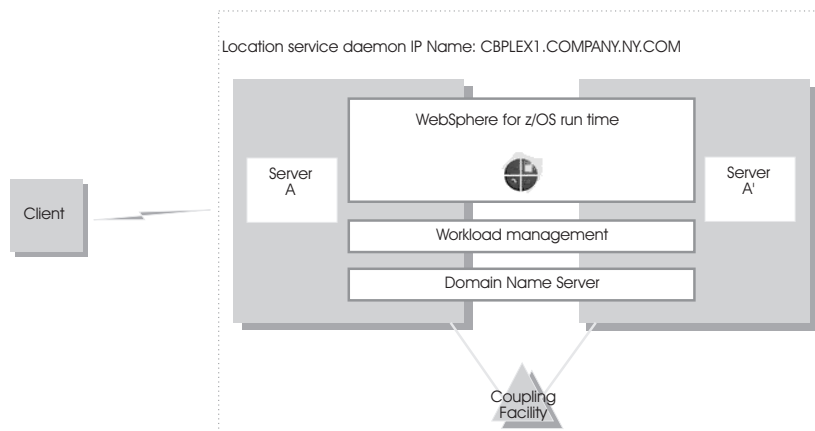
Workload management provides the following benefits to applications that are running on an application server:

- It balances server workloads, allowing processing tasks to be distributed according to the capacities of the different machines in the sysplex.
- It provides failover capability by redirecting client requests if one or more servers is unable to process them. This improves the availability of applications and administrative services.
- It enables systems to be scaled up to serve a higher client load than provided by the basic configuration. With clustering, additional instances of servers, servlets, and other objects can easily be added to the configuration.
- It enables servers to be transparently maintained and upgraded while applications remain available for users.
- It centralizes the administration of servers and other objects.

In the product environment, you implement workload management by using clusters, transports, and replication domains.

Connection optimization

Characteristics of a configuration in which the Domain Name Server cooperates with workload management (WLM) to route client requests throughout a cell are:



- The domain name server (DNS) is replicated by setting up a secondary DNS on more than one system in the cell.
- The client must know the host name and port of the name server to connect to WebSphere Application Server for z/OS.
- Each system in the cell has the same location service daemon IP name. Workload management and the domain name server determine the actual system that receives client requests. The client sees the cell as a single system, though its requests might be balanced across systems in the cell.

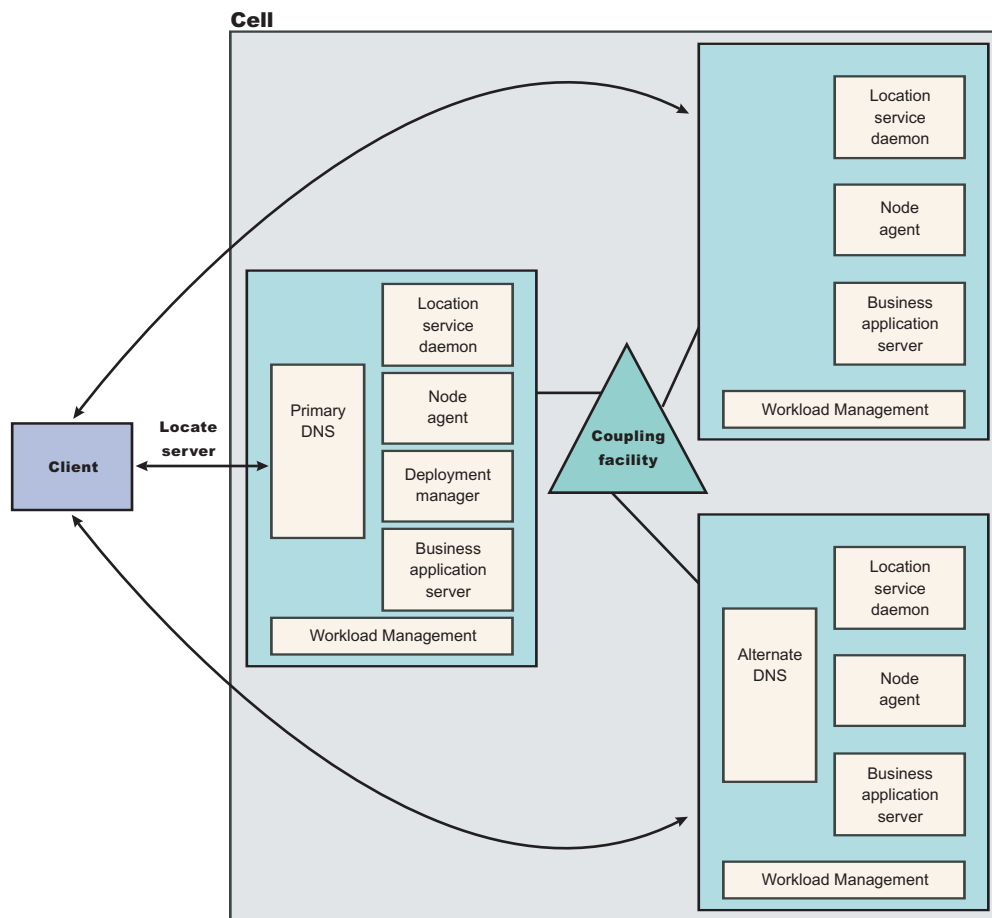
- As part of workload balancing and maximizing performance goals, workload management also routes work requests to systems in the cell. This function is possible because WebSphere Application Server for z/OS cooperates with workload management. Because the system references that a client sees are indirect, even requests from that same client might be answered by differing systems in the cell.
- The implication for clients is that they should not cache IP addresses unless they can recover from failed connections. That is, if a connection fails, a client should be able to reissue a request, but, because the IP address is an indirect address, a reissue of the request can be answered by another system in the cell.

For additional details on setting up servers for connection optimization, see *z/OS Communications Server: IP Configuration Reference*.

Sysplex routing of work requests

The product uses the domain name server (DNS) to route work requests within a cell. You can use the DNS, instead of a sysplex distributor to distribute workload and balance requests for the same hostname across multiple IP addresses (one per daemon).

The DNS accepts a generic hostname from the client and maps the name to a specific system. The DNS works with workload management (WLM) to select the best available system. Workload management analyzes the current state of the cell and considers a number of factors, such as CPU, memory, and I/O utilization, when it determines the best system to handle new work. The DNS then routes the client request to that system. This use of workload management and the DNS is optional. However, using workload management and the DNS eliminates a single point of failure.



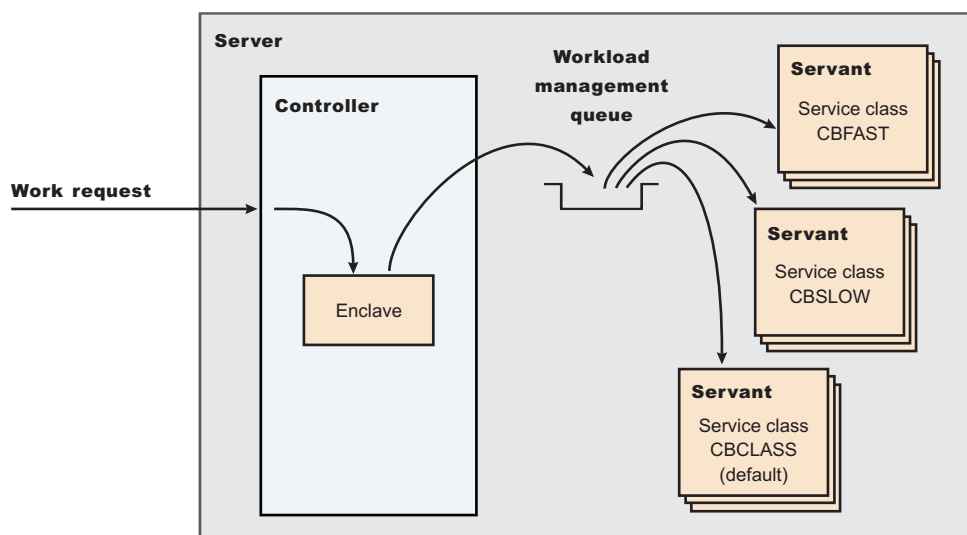
Each system in a cell has the product runtime. All the systems contain a location service daemon, node agent, and business application servers. One system acts as the deployment manager for the cell. The client uses the CORBA General Inter-ORB Protocol (GIOP) to send requests to the product. The location service daemon acts as a location service agent. It accepts locate requests with object keys in the requests. The location service daemon extracts the server cluster name from the object key, and then gives the server name to workload management. Workload management chooses the optimal server in the cell to handle the request. The location service daemon merges specific Interoperable Object Reference (IOR) information that is related to the chosen server with object key information stored in the original IOR. The result of this merging is a direct IOR that gets returned to the client. The client ORB uses this returned reference to establish the IOR connection to the server holding the object of interest.

The transport mechanism that the product uses depends on whether the client is local or remote. A client that is not running on the same z/OS system as the application server, is called a remote client, and requires a TCP/IP transport. If the client is local, the transport is through a program call. Local transport is faster because it does not require a physical trip over the network, eliminates data transforms, simplifies the marshalling of requests, and uses optimized Resource Access Control Facility (RACF) facilities for security rather than having to invoke Kerberos or the Secure Sockets Layer (SSL).

Address space management for work requests

The product propagates the performance context of work requests by using workload management (WLM) enclaves. Each transaction has its own enclave and is managed according to its service class.

The controller of a server, which workload management views as a queue manager, uses the enclave associated with a client request to manage the priority of the work. If the work has a high priority, workload management can direct the work to a high-priority servant in the server. If the work has a low priority, workload management can direct the work to a low-priority servant. The effect is to partition the work according to priority within the same server.



Enclaves can originate in several ways:

- The product uses its own set of rules to create an enclave for a client request from the network.
- Some subsystems, such as the IBM HTTP Sever, create enclaves and pass them to the application server, which, in turn, passes the enclaves on.
- The product treats batch jobs as if they were remote clients.

To communicate the performance context to workload management, you must classify the workloads in your system according to the following work qualifiers.

Table 7. WLM work qualifiers and corresponding product entities

Work qualifier abbreviation	Work qualifier	Corresponding product entity
CN	Collection name	Cluster name
UI	User ID	User ID under which work is running

For more information about classification rules and workload qualifiers, refer to the topic *Classifying z/OS workload* and the z/OS publication *z/OS MVS Planning: Workload Management*.

In addition to client workloads, you must consider the performance of the product run-time servers and your business application servers. In general, server controllers act as work routers, so they must have high priority. Because workload management starts and stops servants dynamically, servants also need high priority to be initialized quickly. After the servants are initialized, they run work according to the priority of the client enclave, so the servant priority that you assign has no significance after initialization.

In summary, use the following table to set the performance goals for each class:

Table 8. Workload management rules

If you are classifying the assign it to:	Explanation
Location service daemon	SYSSTC or a high velocity, high importance STC	The system treats it as a started task, and it must route work requests quickly.
Controller	SYSSTC or a high velocity, high importance STC	A controller must route work quickly, but you must balance the priority of your business application server with other work in the system.
Servant	A lower velocity and importance STC than the controller	If too high, you spend too much time in Java garbage collection, and you might consume more system resources than you want. If too low, JSP compiles and Java garbage collection could delay processing in your application. Startup of additional servant address spaces might also be delayed.
Application environment	Use the CB classification rules, the percentage response time goal, for example 80% of transactions complete in .25 seconds.	
Client applications	Assuming a long-running application, a velocity goal should be used that is relative to other work on the system.	

Example of classification rules

In this example, all work for BBOC001, except for work running under the user ID DBOOZ, gets classified as CBFAST. Work for DBOOZ gets classified as CBSLOW. All other work, such as work coming from clients outside the cell and including the work for the product runtime servers, gets classified as CBCLASS.

Purpose of this example

Let us assume you have three workload management service classes defined for the product(subsystem type CB):

1. CBFAST-designed for transactions requiring fast response times.

2. CBSLOW-designed for long-running applications that do not require fast response times.
3. CBCLASS-designed for remaining work requests.

You design a client workload called BBOC001 that requires fast response times. Also, you want to give work that runs under your manager's user ID (DBOOZ) slower response times. Finally, all remaining work requests should run under the default service class, CBCLASS.

Table 9. Classification rules example

Type column	Name column	Service column	Goal
CN	BBOC001	CBFAST	90% complete in 2 seconds
UI	DBOOZ	CBSLOW	Velocity 50, importance = 3
(default)	(blank)	CBCLASS	Discretionary

You could set the following performance goals through IWMARIN0:

1. Issue IWMARIN0 and choose option 4:

```

File Utilities Notes Options Help
-----
Functionality LEVEL003  Definition Menu  WLM Appl LEVEL004  Command ==>
-----

Definition data set . . . : 'CB.MYCB.WLM'
Definition name . . . . . CB390      (Required)
Description . . . . . WLM Setup for the product
Select one of the following options. . . . . 4__
1. Policies
2. Workloads
3. Resource Groups
4. Service Classes
5. Classification Groups
6. Classification Rules
7. Report Classes
8. Service Coefficients/Options
9. Application Environments
10. Scheduling Environments

```

2. Create a service class called CBFAST and specify that it be 90% complete in 2 seconds.

Note: The example assumes you have defined a workload called ONLINE.

```

Service-Class Notes Options Help
-----
Create a Service Class
Row 1 to 2 of 2  Command ==>
Service Class Name . . . . . CBFAST  (Required)
Description . . . . . Quick CB transactions
Workload Name . . . . . ONLINE    (name or ?)
Base Resource Group . . . . .      (name or ?)
Specify BASE GOAL information.  Action Codes: I=Insert new period,
E=Edit period, D=Delete period.
---Period---  -----Goal-----
Action #  Duration  Imp.  Description
-----
1          1      90% complete within 00:00:02.000
***** Bottom of data *****

-----
| Press EXIT to save your changes or CANCEL to discard them. (IWMAM970) |
|-----|

```

3. Save the service class. You see the following:

Service-Class View Notes Options Help

```
-----
Service Class Selection List
Row 1 to 14 of 21  Command ==>
Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
/=Menu Bar
Action Class
Description
Workload
_____ CBFAST
Quick CB Transactions
ONLINE
***** Bottom of data *****
```

- 4. Repeat these steps for the CBSLOW service class.
- 5. Create classification rules using the new service class. Choose option 6 on the main panel:

File Utilities Notes Options Help

```
-----
Functionality LEVEL003          Definition Menu          WLM Appl LEVEL004
Command ==>
Definition data set . . . : 'CB.MYCB.WLM'
Definition name . . . . . CB390 (Required)
Description . . . . . WLM Setup for the product
Select one of the following options. . . . . 6__
1. Policies
2. Workloads
3. Resource Groups
4. Service Classes
5. Classification Groups
6. Classification Rules
7. Report Classes
8. Service Coefficients/Options
9. Application Environments
10. Scheduling Environments
```

- 6. Create a set of rules for your service classes:

Subsystem-Type Xref Notes Options Help

```
-----
Create Rules for the Subsystem Type          Row 1 to 2 of 2
Command ==>                                SCROLL ==> PAGE
Subsystem Type . . . . . CB (Required)
Description . . . . . WebSphere classification
Fold qualifier names? . . . . Y (Y or N)
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
-----Qualifier-----
-----Class-----
Action Type Name Start
Service Report
DEFAULTS: CBCLAS
_____ 1 CN
BBOC001 _____
CBFAST _____
_____ 1 UI
DB00Z _____
CBSLOW _____
***** BOTTOM OF DATA *****
```


Related tasks

“Using transaction classes to classify workload for WLM” on page 320

You can use transaction classes to classify client workload for workload management (WLM). The workload that WLM manages consists of different transactions that are targeted to separate servants, each with goals defined by specific service classes. The service classes chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive. You do not want to set a servant higher in the service class hierarchy than more important work such as production WebSphere, CICS, or IMS transaction servers.

Enabling multiple servants on z/OS

Use this task to enable multiple servants on your system. Enabling multiple servants improves the performance of the product on z/OS..

Before you begin

Review the limitations of enabling multiple servants.

About this task

Workload Management (WLM), enables you to manage the performance and the number of servants that are running.. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active. With this task, you can enable WLM to start additional servants to meet performance goals.

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***.
2. Under Server Infrastructure, click **Java and process management > Server instance**.
3. Select the Multiple instances enabled field.
4. Click **Apply** to finish the Server Instance changes.
5. Click **OK**, and then click **Save** to save your configuration changes.

Results

Workload Management (WLM) can start additional servant regions to meet performance goals, based on the importance of its work compared to other work in the system, the availability of system resources that are needed to satisfy those objectives, and a determination by WLM of whether starting more address spaces might help achieve the objectives. It is also important to make the goals reasonable.

Multiple servant regions

With workload management (WLM), you can control the number of servants that are running and how they are performing. WLM manages the response time and throughput of transactions according to their assigned service class, the associated performance objectives, and the availability of system resources. To meet these goals, WLM sometimes needs to control or override the number of servants that are active.

Product applications are deployed within a generic *server*. One or more server instances must be defined on one or more systems within a *node*. Each server instance consists of a *controller* and one or more *servants*. The controllers are started by MVS as *started tasks*, and servants are started by WLM, as they are needed.

WLM dynamic application environments can be enabled on your z/OS system if you are running on z/OS Release 2, with the service update for APAR OW54622 installed, or if you are running on a higher level of z/OS. If you have WLM dynamic application environments enabled on your system, WLM honors the specifications for the number of servant regions. If you are using static application environments, which is

a setting that is specified through the WLM ISPF panels, then you must also enable multiple servant regions by indicating No limit in the WLM ISPF panels.

Note: If you use multiple servants, remember that:

- The administrative console and the deployment manager no longer have serialization requirements that only work in a single Java virtual machine (JVM). Therefore, you can run these applications in a server with multiple servants.
- If you specify a maximum number of instances, workload management (WLM) cannot start more than the specified number of servant regions for this server instance.
- The maximum number of servants should be at least as large as the number of different service classes that might be used by transactions that are run in the server. The number must also account for the *default CB-type service class* and enclaves that might originate outside of the product servers and are classified by other classification rules, such as the IBM HTTP Server (IHS).

Controlling the number of servants

You can control the minimum or maximum number of servants for a server using the administrative console. The minimum value is useful for starting up a basic number of servants before your work arrives. This can reduce delays while waiting for the workload manager (WLM) to start up additional servants.

About this task

The maximum value is useful for *capping* the number of address spaces that are started by WLM for each *server instance*, if you determine that excessive servant regions are contributing to service degradation.

1. Start the administrative console.
2. Click **Servers > Server Types > WebSphere application servers > server_name**.
3. Under Server Infrastructure, click **Java and process management > Server instance**
4. Type a value into the **Minimum Number of Instances** and **Maximum Number of Instances** fields, or leave these fields blank to allow WLM to determine the numbers.
5. Click **Apply** to finish the Server Instance changes.
6. Click **Save** to save your changes.

Classifying z/OS workload

You can use a common workload classification document to classify inbound HTTP, IIOp, and message-driven bean (MDB) work requests for the z/OS workload manager.

Before you begin

You should use workload management on a z/OS system. See “Workload management (WLM) for z/OS” on page 296 for more information.

About this task

A workload classification document file is an XML file in which you classify incoming HTTP, IIOp, and message-driven bean (MDB) work requests and assign them to a transaction class (TCLASS). The TCLASS value, if it is assigned, is passed to the MVS Workload Manager. WLM uses the TCLASS value to classify the inbound work requests and to assign a service class or a report service class to each request.

The common workload classification document is the method you should use to classify work requests in a z/OS environment. Support for other WebSphere Application Server mechanisms for classifying work in a z/OS environment is deprecated and you should no longer use those mechanisms.

If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to define a Classification element that uses SibClassification elements. You must also perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS". If you replace any listener port with a JMS activation specification for use by MDB applications with the Version 6 default messaging provider, you should replace any related InboundClassification type="mdb" classifications with SibClassifications type="jmsra" classifications.

If you want to classify work for message-driven beans deployed against a WebSphere MQ messaging provider activation specification, you need to define a Classification element that uses WMQRAClassification elements. You must also perform z/OS Workload Manager actions that are required to use the TCLASS value "WMQRA". If you replace any listener port with a JMS activation specification for use by MDB applications with the WebSphere MQ messaging provider, you should replace any related InboundClassification type="mdb" classifications with WMQRAClassification classifications.

1. Develop the workload classification document. Use the information in the article, "Workload classification file" on page 310. See "Sample z/OS workload classification document" on page 307 for a sample of the workload classification document and the DTD.
2. If you create the document on a z/OS system in codepage IBM-1047, the normal codepage for files that exist in the HFS, you must convert the file to ASCII before you use the file. Use one of the following options to convert a working document into a document that can be used by the server:

- native2ascii

This is a utility in the Java SDK that can convert a file from the native codepage to the ASCII codepage. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

```
/u/userid -> native2ascii \  
x5sr02.classification.ebcdic.xml > x5sr02.classification.xml
```

The command line is split with the backslash (\) character to the next line for publication purposes.

- iconv

This is a z/OS utility that can convert files from one designated codepage to a different designated codepage. For example, if you are working on an XML document called x5sr02.classification.ebcdic.xml and you want to create a document called x5sr02.classification.xml, use the following command:

```
/u/userid -> iconv -f IBM-1047 -t UTF-8 \  
x5sr02.classification.ebcdic.xml > x5sr02.classification.xml
```

The command line is split with the backslash (\) character to the next line for publication purposes.

- Create the document on your workstation and then FTP the file to the correct location on the z/OS system in binary format. By using this option, you can also create the Classification.dtd file in the same directory as the workload classification document. Then, you can perform an XML validity check on the document before installing it on a server. Use any type of validating parser, for example, you can use WebSphere Application Developer workbench to construct and validate the workload classification document.
3. Specify the location of the workload classification document in the administrative console. Use the wlm_classification_file variable to specify the XML file that contains the classification information. In the administrative console, click **Environment > WebSphere variables > New**. You can set the variable at cell, node, or server instance level. If you specify the variable at the cell or node level, the information must be accessible and applicable to all the servers that inherit the specification from the node or cell.
 4. You must perform z/OS Workload Manager actions that are required to use the TCLASS values. Each TCLASS must be assigned a service class, report service class, or both to the enclave under which the work runs. The CB classification rules must be updated.

If you want to classify work for message-driven beans deployed against JCA 1.5 resources with the default messaging provider, or you want to classify mediation work for use with service integration buses, you need to perform z/OS Workload Manager actions that are required to use the TCLASS value "SIBUS".

Transaction classes are used as sub-rules in establishing service classes and transaction. The TCLASS values are not used as level one rules. If you decide to use TCLASS as a level one rule rather than a sub-rule, you must be careful in ordering the rules. The first level one rule that applies to the work is used, so more specific rules should be first, followed by the broad rules. For example, consider the following two examples of CB classification rules:

```
Subsystem-Type Xref Notes Options Help
-----
Modify Rules for the Subsystem Type Row 1 to 17 of 17
Command ==> _____ SCROLL ==> CSR
Subsystem Type . : CB Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSES
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ==>
      -----Qualifier-----          -----Class-----
Action  Type  Name  Start          Service Report
      DEFAULTS: CBCLASS RWASDEF
_____ 1  CN  P5SR01*  1          CBCLASS RTP5CLUS
_____ 1  TC  A0          _____ CBHUTCH RP5A0
_____ 1  TC  A1          _____ CBHUTCH RP5A1
_____ 1  TC  A1B         _____ CBHUTCH RP5A1B
_____ 1  CN  WSIVP2*    _____ CBSLOW RWSIVP
_____ 1  CN  T%SERV*    1          CBFAST RTSMIGT
_____ 1  CN  B4*        _____ CBFAST _____
```

In the preceding example, the TCLASS assignments that are made for enclaves running in the server P5SR01x are never used by the workload manager. When the following rule is run, no further searching of the classification table is done:

```
_____ 1  CN  P5SR01*  1          CBCLASS
```

The TCLASS assignments are not used. All of the enclaves that run in the P5SR01x servers are assigned to the CBCLASS service class and the RTP5CLUS report service class.

```
Subsystem-Type Xref Notes Options Help
-----
Modify Rules for the Subsystem Type Row 1 to 17 of 17
Command ==> _____ SCROLL ==> CSR
Subsystem Type . : CB Fold qualifier names? Y (Y or N)
Description . . . CB Class'n w/WLM Trans. CLASSES
Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ==>
      -----Qualifier-----          -----Class-----
Action  Type  Name  Start          Service Report
      DEFAULTS: CBCLASS RWASDEF
_____ 1  TC  A0          _____ CBHUTCH RP5A0
_____ 1  TC  A1          _____ CBHUTCH RP5A1
_____ 1  TC  A1B         _____ CBHUTCH RP5A1B
_____ 1  CN  P5SR01*    1          CBCLASS RTP5CLUS
_____ 1  CN  WSIVP2*    _____ CBSLOW RWSIVP
_____ 1  CN  T%SERV*    1          CBFAST RTSMIGT
_____ 1  CN  B4*        _____ CBFAST _____
```

In the preceding example, if a TCLASS value of A0, A1, or A1B are provided in the classification, they are used regardless of which server is running the work. In this case, the server name is used only if these three TCLASS values are not present.

- Restart the application server to implement any changes to the file. If the workload classification document is not a well formed, valid XML document it is ignored by the application server and the following message is displayed:

BB0J0085E PROBLEMS ENCOUNTERED PARSING WLM CLASSIFICATION XML FILE (0)

6. Use the DISPLAY WORK operator command to display classification information. Use this command to determine if your classification scheme is classifying the work as you intended. Issue the following command to display the IIOF and HTTP classification information:

```
MODIFY|F <servername>, DISPLAY,WORK,CLINFO
```

Issue this command against each application server.

The following example shows a possible result of issuing the new operator command:

```
F X5SR02A,DISPLAY,WORK,CLINFO
```

```
BB000281I CLASSIFICATION COUNTERS FOR IIOF WORK
BB000282I CHECKED 14, MATCHED 14, USED 0, COST 0, DESC: IIOF Default
BB000282I CHECKED 14, MATCHED 3, USED 0, COST 0, DESC: sample
BB000282I CHECKED 3, MATCHED 1, USED 1, COST 3, DESC: a1a
BB000282I CHECKED 2, MATCHED 1, USED 1, COST 4, DESC: a1b
BB000282I CHECKED 1, MATCHED 1, USED 1, COST 5, DESC: a1c
BB000282I CHECKED 11, MATCHED 11, USED 0, COST 0, DESC: other
BB000282I CHECKED 11, MATCHED 1, USED 1, COST 4, DESC: a
BB000282I CHECKED 10, MATCHED 1, USED 1, COST 5, DESC: b
BB000282I CHECKED 9, MATCHED 1, USED 1, COST 6, DESC: c
BB000282I CHECKED 8, MATCHED 2, USED 2, COST 7, DESC: d
BB000282I CHECKED 6, MATCHED 1, USED 1, COST 8, DESC: e
BB000282I CHECKED 5, MATCHED 4, USED 4, COST 9, DESC: f
BB000282I CHECKED 1, MATCHED 1, USED 1, COST 10, DESC: g
BB000283I FOR IIOF WORK: TOTAL CLASSIFIED 14, WEIGHTED TOTAL COST 95
BB000281I CLASSIFICATION COUNTERS FOR HTTP WORK
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: HTTP Default
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: n
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: o
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: q
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: r
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: s
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: p
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: t
BB000283I FOR HTTP WORK: TOTAL CLASSIFIED 0, WEIGHTED
BB000281I CLASSIFICATION COUNTERS FOR INTERNAL WORK
BB000282I CHECKED 76, MATCHED 76, USED 76, COST 1, DESC: Internal Default
BB000283I FOR INTERNAL WORK: TOTAL CLASSIFIED 76, WEIGHTED TOTAL COST 76
BB000281I CLASSIFICATION COUNTERS FOR SIP WORK
BB000282I CHECKED 0, MATCHED 0, USED 0, COST 0, DESC: SIP Default
BB000283I FOR SIP WORK: TOTAL CLASSIFIED 0, WEIGHTED TOTAL COST 0
BB000188I END OF OUTPUT FOR COMMAND DISPLAY,WORK,CLINFO
```

An explanation of the command output follows:

BB000281I CLASSIFICATION COUNTERS FOR *type* WORK

The header message for messages that display the usage of the workload classification rules. The value of *type* can be HTTP or IIOF. You cannot display inbound MDB classifications.

BB000282I CHECKED *n1*, MATCHED *n2*, USED *n3*, COST *n4*, DESC: *text*

This message displays information about a particular rule in the workload classification. This message displays the following information:

- *n1* - The number of times the rule has been examined.
- *n2* - The number of this times that this rule has been matched by the request.
- *n3* - The number of times that this rule has actually been used.
- *n4* - The cost of using the rule, or the number of compares that are required to determine if this is the correct rule to use.
- *text* - The descriptive text from the classification rule so that you can tell which classification rule is being displayed.

BB000283I FOR *type* WORK: TOTAL CLASSIFIED *n1*, WEIGHTED TOTAL COST *n2*

This message shows the summary information for the HTTP or IIOp work classification. This message displays the following information:

- *type* - The type of work that is being displayed. The value must be IIOp or HTTP.
- *n1* - The number of requests that were classified using the classification rules.
- *n2* - The weighted total cost, calculated by taking the number of times that each rule was used multiplied by the cost, or number of rule compares that were done, of using the rule and adding those up across all of the rules.

The total cost *n2* divided by the total number of requests classified *n1* equals the cost of using the table. The closer that the value is to one, the lower the cost of using the defined rules. A value of 1 indicates that there is just the default classification, so no requests match it.

7. Repeat these steps until you achieve your optimal workload distribution and costs.

Results

The workload classification document is used to classify inbound requests.

Sample z/OS workload classification document

Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Classification SYSTEM "Classification.dtd" >
<Classification schema_version="1.0">
<!--
      Internal Classification Rules
-->
  <InboundClassification type="internal"
                        schema_version="1.0"
                        default_transaction_class="value1"/>
</InboundClassification>

<!--
      IIOp Classification Rules
-->
  <InboundClassification type="iioP"
                        schema_version="1.0"
                        default_transaction_class="A0">
    <iioP_classification_info transaction_class="A1"
                            application_name="IIOpStatelessSampleApp"
                            module_name="StatelessSample.jar"
                            component_name="Sample20"
                            description="Sample20 EJB Classification">
      <iioP_classification_info transaction_class=""
                              method_name="echo"
                              description="No TCLASS for echo()" />
      <iioP_classification_info transaction_class="A1B"
                              method_name="ping"
                              description="Ping method" />
    </iioP_classification_info>
    <iioP_classification_info application_name="*"
                              module_name="*"
                              component_name="*"
                              transaction_class="A2"
                              description="TCLASS the rest to A2">
      <iioP_classification_info transaction_class="A2A"
                              method_name="resetFilter"
                              description="Sp1 case resetFilter()" />
    </iioP_classification_info>
  </InboundClassification>

<!--
```

```

    HTTP Classification Rules
-->
<InboundClassification type="http"
    schema_version="1.0"
    default_transaction_class="M">
  <http_classification_info transaction_class="N"
    host="yourhost.yourcompany.com"
    description="Virtual Host yourhost">
    <http_classification_info transaction_class="0"
      port="9080"
      description="Def yourhost HTTP reqs">
      <http_classification_info transaction_class="Q"
        uri="/gcs/admin"
        description = "Gcs" />
      <http_classification_info transaction_class="S"
        uri="/gcs/admin/1*"
        description="GCS login" />
    </http_classification_info>
    <http_classification_info transaction_class="P"
      port="9081"
      description=" Def yourhost HTTPS reqs "/>
    <http_classification_info transaction_class=""
      uri="/gcsmgr/*"
      description="GCSS Mgr" />
  </http_classification_info>
</http_classification_info>
</InboundClassification>

<!--
    SIP Classification Rules
-->
<InboundClassification type="sip"
    schema_version="1.0"
    default_transaction_class="value1"/>
</InboundClassification>

<!--
    MDB Classification Rules
-->
<InboundClassification type="mdb"
    schema_version="1.0"
    default_transaction_class="qrs">
  <endpoint type="messageListenerPort"
    name="IVPListenerPort"
    defaultClassification="MDBX"
    description="ABC">
    <classificationentry selector="Location=&apos;East&apos;"
      classification="MDB1"
      description="DEF"/>
    <classificationentry selector="Location< >&apos;East&apos;"
      classification="MDB2"
      description="XYZ" />
  </endpoint>
  <endpoint type="messageListenerPort"
    name="SimpleMDBListenerPort"
    defaultClassification="MDBX"
    description="GHI" />
</InboundClassification>

  <SibClassification type="jmsra" schema_version="1.0"
    default_transaction_class="a">
    <sib_classification_info transaction_class="b"
      selector="user.Location=&apos;East&apos;" bus="magic"
      destination="nowhere" description="n" />
    <sib_classification_info transaction_class="c"
      selector="user.Location=&apos;West&apos;" bus="omni" description="n" />
  </SibClassification>

```



```

<SibClassification type="destinationmediation" schema_version="1.0"
  default_transaction_class="b">
  <sib_classification_info transaction_class="e"
    selector="user.Location=&apos;East&apos;" destination="themoon"
    discriminator="sides/dark" description="n" />
  <sib_classification_info transaction_class="f"
    selector="user.Location=&apos;West&apos;" description="n" />
</SibClassification>

<WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
  <wmqra_classification_info transaction_class="TC_1"
    queue_manager="GOLD"
    description="gold queue manager maps to TC_1"/>
  <wmqra_classification_info transaction_class="TC_2"
    selector="JMSPriority&gt;7"
    description="high priorities maps to TC_2"/>
  <wmqra_classification_info transaction_class="TC_3"
    selector="JMSPriority&gt;3 AND JMSPriority&lt;8"
    description="medium priorities maps to TC_3"/>
</WMQRAClassification>

<!--
Workload Classification Document for P5SR01x servers
Change History
-----
Activity          Date          Author
Created          01-28-2005  IPL
--->
</Classification>

```

DTD for the workload classification XML document

```

<?xml version='1.0' encoding="UTF-8"?>
<ELEMENT Classification (InboundClassification|SibClassification|WMQRAClassification)+>
<!ATTLIST Classification schema_version CDATA #REQUIRED>
<ELEMENT InboundClassification ((iioop_classification_info*|http_classification_info*|endpoint*))>
<!ATTLIST InboundClassification type (iioop|mdb|http|internal|sip) #REQUIRED>
<!ATTLIST InboundClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST InboundClassification schema_version CDATA #REQUIRED>
<ELEMENT iioop_classification_info (iioop_classification_info*)>
<!ATTLIST iioop_classification_info activity_workload_classification CDATA #IMPLIED>
<!ATTLIST iioop_classification_info application_name CDATA #IMPLIED>
<!ATTLIST iioop_classification_info component_name CDATA #IMPLIED>
<!ATTLIST iioop_classification_info description CDATA #IMPLIED>
<!ATTLIST iioop_classification_info method_name CDATA #IMPLIED>
<!ATTLIST iioop_classification_info module_name CDATA #IMPLIED>
<!ATTLIST iioop_classification_info transaction_class CDATA #REQUIRED>
<ELEMENT endpoint (classificationentry*)>
<!ATTLIST endpoint defaultclassification CDATA #REQUIRED>
<!ATTLIST endpoint name CDATA #REQUIRED>
<!ATTLIST endpoint type (messagelistenerport) #REQUIRED>
<!ATTLIST endpoint description CDATA #IMPLIED>
<ELEMENT classificationentry EMPTY>
<!ATTLIST classificationentry classification CDATA #REQUIRED>
<!ATTLIST classificationentry selector CDATA #REQUIRED>
<!ATTLIST classificationentry description CDATA #IMPLIED>
<ELEMENT http_classification_info (http_classification_info*)>
<!ATTLIST http_classification_info host CDATA #IMPLIED>
<!ATTLIST http_classification_info port CDATA #IMPLIED>
<!ATTLIST http_classification_info uri CDATA #IMPLIED>
<!ATTLIST http_classification_info description CDATA #IMPLIED>
<!ATTLIST http_classification_info transaction_class CDATA #REQUIRED>
<ELEMENT SibClassification (Sib_classification_info+)>
<!ATTLIST SibClassification type (jmsra|destinationmediation) #REQUIRED>
<!ATTLIST SibClassification default_transaction_class CDATA #REQUIRED>

```

```

<!ATTLIST SibClassification schema_version CDATA #REQUIRED>
<!ELEMENT sib_classification_info EMPTY>
<!ATTLIST sib_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST sib_classification_info selector CDATA #IMPLIED>
<!ATTLIST sib_classification_info bus CDATA #IMPLIED>
<!ATTLIST sib_classification_info destination CDATA #IMPLIED>
<!ATTLIST sib_classification_info discriminator CDATA #IMPLIED>
<!ATTLIST sib_classification_info description CDATA #IMPLIED>
<!ELEMENT WMQRAClassification (wmqra_classification_info+)>
<!ATTLIST WMQRAClassification default_transaction_class CDATA #REQUIRED>
<!ATTLIST WMQRAClassification schema_version CDATA #REQUIRED>
<!ELEMENT wmqra_classification_info EMPTY>
<!ATTLIST wmqra_classification_info transaction_class CDATA #REQUIRED>
<!ATTLIST wmqra_classification_info selector CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info queue_manager CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info destination CDATA #IMPLIED>
<!ATTLIST wmqra_classification_info description CDATA #IMPLIED>

```

Workload classification file

The workload classification document is a common XML file that classifies inbound HTTP, IIOp, message-driven bean (MDB), and mediation work for the z/OS workload manager.

Usage notes

You must perform the task “Classifying z/OS workload” on page 303 when you use the workload classification document. See “Sample z/OS workload classification document” on page 307 for an example of a workload classification document.

Required elements

<?xml version="1.0" encoding="UTF-8"?>

Indicates that the workload classification document must be saved in ASCII to be processed by the application server. This statement is required.

<!DOCTYPE Classification SYSTEM "Classifications">

Provides the XML parser with the name of the DTD document that is provided with the product, and that is used to validate the workload classification document. The workload classification document that you write must follow the rules that are described in this DTD. You must add this statement to the workload classification document.

Classification

<Classification schema_version="1.0">

Indicates the root of the workload classification document. Every workload classification document must begin and end with this element. The `schema_version` attribute is required. The only supported `schema_version` is 1.0. The `Classification` element contains one or more `InboundClassification` elements. For inbound service integration work, the `Classification` element can also contain up to two `SibClassification` elements. If classifying inbound messages for delivery to message-driven beans using WebSphere MQ messaging provider activation specifications, the `Classification` element can contain one or more `WMQRAClassification` elements.

InboundClassification

<InboundClassification type="iiop | http | mdb" schema_version="1.0" default_transaction_class="value">

Use the following rules when using the `InboundClassification` element:

- The **type** attribute is required. The value must be `internal`, `iiop`, `http`, `mdb`, or `sip`. Only one occurrence of an `InboundClassification` element can occur in the document for each type. There can be up to five `InboundClassification` elements in a document. The types do not have to be specified in a certain order in your classification document.
- The **schema_version** attribute is required. The value must be set to 1.0.

- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").
- The InboundClassification elements cannot be nested. Each InboundClassification element must end before the next InboundClassification element or SibClassification element can begin.

SibClassification

```
<SibClassification type="jmsra | destinationmediation" schema_version="1.0"
default_transaction_class="value">
```

Use the following rules when using the SibClassification element:

- The **type** attribute is required. The value must be jmsra or destinationmediation. There can be at most one SibClassification element in the document for each type. The types do not have to be specified in a certain order in your classification document.
- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " ").
- The SibClassification elements cannot be nested. Each SibClassification element must end before the next InboundClassification element or SibClassification element can begin.

WMQRAClassification

```
<WMQRAClassification schema_version="1.0" default_transaction_class="value">
```

The following rules apply to the WMQRAClassification element:

- The **schema_version** attribute is required. The value must be set to 1.0.
- The **default_transaction_class** attribute must be specified, and defines the default transaction class for work flows of the specified type. The string value must be a valid WLM transaction class.
- The WMQRAClassification elements cannot be nested. Each WMQRAClassification element must end before any other classification elements can begin.

The rules and XML statements for classifying different types of work are very similar, but there is slightly different syntax for each type. For more information about the syntax for each type of work, see the following sections:

InboundClassification

- "Internal Classification" on page 312
- "IIOP Classification" on page 312
- "HTTP classification" on page 314
- "MDB classification" on page 315
- "SIP Classification" on page 316

SibClassification

- "JMS RA classification" on page 316
- "Mediation classification" on page 318

WMQRAClassification

- "WebSphere MQ messaging provider classification" on page 319

Internal Classification

The InboundClassification element with the attribute type="internal" defines the section of the document that is applicable to internal work, such as requests that are dispatched in a servant, that originate in the owning controller. An example of this element follows:

```
<InboundClassification type="internal" schema_version="1.0"
    default_transaction_class="value1">
```

If an InboundClassification element with the type="internal" attribute is not specified, internal work is classified using the rules that are specified for IOP work.

IOP Classification

The InboundClassification element with the attribute type="iop" defines the section of the document that is applicable to IOP classification. An example of this element follows:

```
<InboundClassification type="iop" schema_version="1.0"
    default_transaction_class="value1">
```

You can classify IOP work based on the following Java Platform, Enterprise Edition (Java EE) application artifacts:

- Application name
The name of the application that contains the enterprise beans. It is the display name of the application, which might not be the name of the .ear file that contains all of the artifacts.
- Module name
The name of the EJB .jar file that contains one or more enterprise beans. There can be multiple EJB .jar files in an .ear file.
- Component name
The name of the EJB that is contained in a module (or EJB .jar file). There can be one or more enterprise beans contained in a .jar file.
- Method name
The name of a remote method on an EJB.

Classify IOP work in various applications at any of these levels by using the iop_classification_info element.

iiop_classification_info

```
<iop_classification_info transaction_class="value1"
    [application_name="value2"]
    [module_name="value3"]
    [component_name="value4"]
    [method_name="value5"]
    [description="value6"] >
```

With the iop_classification_info element, you can build filters based on the application, module, component, and method names to assign TCLASS values to inbound requests. Use the following rules when using the iop_classification_info element:

- The transaction_class attribute must be specified. The string value must be a valid WLM transaction class, a null string (such as "") or a string that contains eight or fewer blanks (such as " "). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.
- The attributes application_name, module_name, component_name, and method_name can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested iop_classification_info element to assign the transaction class. You can specify the values of these attributes in the following ways:

- The exact name of the application, module, component, or method.
- A wild carded value. You can place an asterisk (*) anywhere in a string to indicate that any string that starts with the string preceding the asterisk and ends with the string that follows the asterisk is considered a match. If the asterisk is at the end of the string, any string that starts with the string preceding the asterisk is considered a match.

Examples:

- The string Mar*61, matches Mar61, March61, and Mar20early61, but does not match March81.
- The string MAR* matches MARCH, Mar61, and MARS, but does not match May61.

You can use any combination of these attributes to make a classification filter. But you should only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the application_name attribute.

- The iiop_classification_info elements can be nested in a hierarchical manner. By nesting the elements, you can create classification filters that are based on the attribute values. The following filter classifies requests on the EJB1 and EJB2 enterprise beans in the MyAPP1 application:

```
<iiop_classification_info transaction_class="FAST"
                        application_name="MyAPP1"
                        component_name="EJB1" />
<iiop_classification_info transaction_class="SLOW"
                        application_name="MyAPP1"
                        component_name="EJB2" />
```

The following filter also classifies requests on EJB1 and EJB2 in the MyAPP1 application, but also classifies requests on any other EJB in the application:

```
<iiop_classification_info transaction_class="MEDIUM"
                        application_name="MyAPP1">
  <iiop_classification_info transaction_class="FAST"
                        component_name="EJB1" />
  <iiop_classification_info transaction_class="SLOW"
                        component_name="EJB2" />
</iio_classification_info>
```

- If you specify an attribute value that conflicts with the parent element's attribute value, the lower level filter is negated. An example of a child value that conflicts with the parent element's attribute value follows:

```
<iiop_classification_info transaction_class="FAST"
                        application_name="MyAPP1">
  <iiop_classification_info transaction_class="SLOW"
                        application_name="MyAPP2" />
</iio_classification_info>
```

In this example, EJB Requests in MyAPP2 would never be assigned to transaction class "SLOW" because the higher level filter only allows IIO requests for application_name="MyAPP1" to be passed through to the lower level filter.

- The first filter at a specific level that matches the attributes of the request is used, not the best or most restrictive filter. Therefore, the order that you specify filters is important.

```
<iiop_classification_info transaction_class="FAST"
                        application_name="MyAPP" />
  <iiop_classification_info transaction_class="SLOW"
                        component_name="*" />
  <iiop_classification_info transaction_class="MEDIUM"
                        component_name="MySSB" />
</iio_classification_info>
```

In the preceding example, all the IIO requests that are processed by enterprise beans in the MyAPP application are assigned a TCLASS value of SLOW. This is true for any requests to the MySSB enterprise as well. Even though MySSB is assigned a transaction class, the filter is not

applied because the first filter was applied and was assigned a TCLASS value of SLOW. The remaining list of filters at the same level is ignored.

- The description field is optional. However, you should use a description on all `iiop_classification_info` elements. The description string prints as part of the operator command support so you can identify the classification rules that are being used. Keep your descriptions reasonably short because they are displayed in the MVS console.

HTTP classification

The `InboundClassification` element with the attribute `type="http"` defines the section of the document that is applicable to HTTP classification. An example of this element follows:

```
<InboundClassification type="http"
    schema_version="1.0"
    default_transaction_class="value1">
```

HTTP work can be classified based on the following J2EE artifacts:

- Virtual host name
Specifies the host name in the HTTP header to which the inbound request is being sent.
- Port number
Specifies the port on which the HTTP catcher is listening.
- URI (Uniform Resource Identifier)
The string that identifies the Web application.

You can classify HTTP work in various applications at any of these levels by using the `http_classification_info` element.

```
<http_classification_info transaction_class="value1"
    [host="value2"]
    [port="value3"]
    [uri="value4"]
    [description="value5"] >
```

With the `http_classification_info` element, you can build filters based on the host, port, and URI to assign TCLASS values to inbound requests. Use the following rules when you use the `http_classification_info` element:

- The `transaction_class` attribute must be specified. The string value must be a valid WLM transaction class, a null string (such as `""`) or a string that contains eight or fewer blanks (such as `" "`). By specifying a null or blank string, you can override a default TCLASS setting, or a TCLASS setting that was assigned by a higher level filter. Specifying a null or blank string means that you do not have a TCLASS value for the request.
- The attributes `host`, `port`, and `uri` can be used as you need them. These attributes act as selectors or filters that either assign a transaction class or allow a nested `http_classification_info` element to assign the transaction class. You can specify the values of these attributes in the following ways:
 - The exact name of the host, port, or uri.
 - A wild carded value. You can use an asterisk (*) to match strings. The string `MAY*` matches any string that starts with `MAY`.
 - Any value. To specify a match to any value, use the asterisk (*) symbol.

Use any or all of these attributes to make a classification filter. Only use the granularity that is required. For example, if there is only one application on the application server, the classification rules do not need to specify the `uri` attribute.

- You can nest the `http_classification_info` elements in a hierarchical manner. You can construct filters based on attribute names. Consider the two following filters:

```

<http_classification_info transaction_class="FAST"
                        host="MyVHost1.com"
                        uri="/MyWebApp1/*" />
<http_classification_info transaction_class="SLOW"
                        host="MyVHost2.com"
                        uri="/MyWebApp2/*" />
<http_classification_info transaction_class="MEDIUM"
                        host="MyVHost1.com">
  <http_classification_info transaction_class="FAST"
                        uri="/MyWebApp1/*" />
  <http_classification_info transaction_class="SLOW"
                        uri="/MyWebApp2/*" />
</http_classification_info>

```

Both filters classify requests to Web applications that are identified by context roots /MyWebApp1 and /MyWebApp2 in the application server that is hosting web applications for virtual host MyVHost1.com. However, the second filter also classifies requests on any other context root in the application server.

- Specifying an attribute name that is different from the parent element's attribute value effectively negates the lower level filter. For example:

```

<http_classification_info transaction_class="FAST"
                        uri="/MyWebApp1/*">
  <http_classification_info transaction_class="SLOW"
                        uri="/MyWebApp2">
  </http_classification_info>
</http_classification_info>

```

This example would never result in Web applications with a context root of /MyWebApp2 being assigned to the transaction class SLOW. The high level filter only allows HTTP requests with a context root of /MyWebApp1/* to be passed to a lower level filter.

- The first filter that is at a specific level is used, not the best or most restrictive filter. Therefore, the order of the filters at each level is important. For example:

```

<http_classification_info transaction_class="FAST"
                        host="MyVHost.com" />
  <http_classification_info transaction_class="SLOW"
                        uri="*" />
  <http_classification_info transaction_class="MEDIUM"
                        uri="/MyWebAppX/*" />
</http_classification_info>

```

In this example, HTTP requests processed by the application server by the virtual host "MyVHost.com" are assigned a TCLASS value of SLOW. Even requests to the Web application with context root /MyWebAppX are assigned a TCLASS value of SLOW because the filter was not applied. The first filter that matches is used for the TCLASS assignment, and the remainder of the filters at the same level is ignored.

- The description field is optional, however, you should use it on all the http_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

MDB classification

The InboundClassification element with the attribute type="mdb" defines the section of the document that applies to work for EJB 2.0 message-driven beans (MDBs) deployed with listener ports. An example of this element follows:

```

<InboundClassification type="mdb"
                        schema_version="1.0"
                        default_transaction_class="qrs">

```


Each InboundClassification element can contain one or more endpoint elements with a messagelistenerport type defined. Define one endpoint element for each listener port that is defined in the server that you want to associate transaction classes with the message-driven bean. An example of the endpoint element follows:

```
<endpoint type="messagelistenerport"
          name="IPVListenerPort"
          defaultclassification="MDBX"
          description="ABC">
```

Use the following rules when defining your endpoint elements:

- The type attribute must always equal messagelistenerport.
- The name attribute corresponds to the listener for your endpoint element. The value of the name attribute must be the name of the listener port that is specified in the administration console for the server.
- The defaultclassification element is the default transaction class that is associated with the message-driven beans. The value of this attribute overrides the default transaction classification value.
- The description field is optional, however, you should use it on all the endpoint elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each endpoint element can have zero, one, or more classificationentry elements. An example of a classification entry element follows:

```
<classificationentry
selector="Location='East'"
          classification="MDB2"
          description="XYZ" />
```

Use the selector attribute of the classificationentry element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your classificationentry elements:

- The value of the selector attribute must match exactly to the selector clause in the MDB deployment descriptor.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references < and >, respectively. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

SIP Classification

The InboundClassification element with the attribute type="sip" defines the section of the document that sets the default transaction class for Session Initiation Protocol (SIP) requests. An example of this element follows:

```
<InboundClassification type="sip" schema_version="1.0"
          default_transaction_class="value1">
```

JMS RA classification

The SibClassification element with the attribute type="jmsra" defines the section of the document that applies to work for message-driven beans (MDBs) deployed against JCA 1.5-compliant resources for use with the JCA resource adapter (RA) of the default messaging provider. An example of this element follows:

```
<SibClassification type="jmsra"
          schema_version="1.0"
          default_transaction_class="a">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info selector="'East'"
    transaction_class="sibb"
    selector="user.Location='East'"
    bus="bigrrred"
    destination="abusqueue"
    description="Some words" />
```

selector

Use the selector attribute of the `sib_classification_info` element to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor. Use the following rules when defining your `sib_classification_info` elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on `SIMessage` messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and `JMS_IBM_properties`) and user properties (which must be prefixed by “.user” - for example, for the user property “Location”, the selector would specify “user.Location” as shown in the preceding example). For more information, see *Working with the message properties*.
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references `<` and `>`, respectively. Similarly, if you use an apostrophe or quotation mark, use the `'` and `"` entity references.

bus

The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination

The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

Although the description field is optional, you should use it on all the `sib_classification_info` elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each `sib_classification_info` element can contain one or more of these properties as needed to classify the work for a message. A `sib_classification_info` element cannot contain more than one instance of each property.

If a message matches several `sib_classification_info` elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info bus="MyBus" transaction_class="a" />
<sib_classification_info destination="MyDest" transaction_class="b" />
```

A message that arrives at destination `MyDest` from the service integration bus `MyBus` is assigned the classification “a”. A message that arrives at `MyDest` from another bus is assigned the classification “b”.

If a message does not match any `sib_classification_info` element in an enclosing `SibClassification` element, the message is assigned the default classification from the `SibClassification` element.

If a message does not match any `sib_classification_info` element in any `SibClassification` element, or if no `SibClassification` elements are defined, all work receives a built-in default classification with the value

“SIBUS”. You must perform z/OS Workload Manager actions that are required to use the TCLASS value “SIBUS”, as described in “Classifying z/OS workload” on page 303.

Mediation classification

The SibClassification element with the attribute type=“destinationmediation” defines the section of the document that applies to work for mediations assigned to destinations on a service integration bus. An example of this element follows:

```
<SibClassification type="destinationmediation"
  schema_version="1.0"
  default_transaction_class="b">
```

Each SibClassification element can contain one or more sib_classification_info elements. An example of a classification entry element follows:

```
<sib_classification_info
  transaction_class="e"
  selector="user.Location=&apos;East&apos;"
  destination="themoon"
  discriminator="sides/dark"
  description="n" />
```

selector

Use the selector attribute of the sib_classification_info element to assign a transaction class to a mediation that has a selector clause in its deployment descriptor. Use the following rules when defining your sib_classification_info elements:

- The value of the selector attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification, but it can operate on SIMessage messages (more than JMS messages). The syntax can select on system properties (including JMS headers, JMSX properties, and JMS_IBM_properties) and user properties (which must be prefixed by “.user” - for example, for the user property “Location”, the selector would specify “user.Location” as shown in the preceding example).
- The value of the selector attribute must have the correct syntax for an XML document. You must replace the < and > symbols with the entity references < and >, respectively. Similarly, if you use an apostrophe or quotation mark, use the ' and " entity references.

bus The name of the service integration bus on which the target destination is assigned. The classification applies to the bus named by this property, or to any bus if you do not specify this property. The destinations to which the classification applies depend on your use of the destination property.

destination

The name of the target bus destination to which the message has been delivered. This is the name of a queue or topic space. The classification applies to the destination named by this property, or any destination if you do not specify this property. The service integration buses to which the classification applies depend on your use of the bus property.

discriminator

The property applies only when the destination property names a topic space. This discriminator value is then an XPath expression that selects one or more topics within the topic space.

description

Although the description field is optional, you should use it on all the sib_classification_info elements. The description is displayed when you monitor the transaction classes in the MVS console.

Each sib_classification_info element can contain one or more of these properties as needed to classify the work for a message. A sib_classification_info element cannot contain more than one instance of each property.

If a message matches several `sib_classification_info` elements, the element that appears first is used. For example, consider the following specifications:

```
<sib_classification_info transaction_class="e" destination="themoon" description="n" />
<sib_classification_info transaction_class="f" description="n" />
```

A message that arrives at the mediated destination `themoon` is assigned the classification “e”. A message that arrives at another mediated destination is assigned the classification “f”.

If a message does not match any `sib_classification_info` element in an enclosing `SibClassification` element, the message is assigned the default classification from the `SibClassification` element.

If a message does not match any `sib_classification_info` element in *any* `SibClassification` element, or if *no* `SibClassification` elements are defined, all work receives a built-in default classification with the value “SIBUS”. You must perform z/OS Workload Manager actions that are required to use the TCLASS value “SIBUS”, as described in “Classifying z/OS workload” on page 303.

WebSphere MQ messaging provider classification

The `WMQRAClassification` element defines the section of the document that applies to work for message-driven beans (MDBs) deployed against WebSphere MQ messaging provider activation specifications. An example of this element follows:

```
<WMQRAClassification default_transaction_class="TC99" schema_version="1.0">
```

A `WMQRAClassification` element can contain one or more `wmqra_classification_info` elements. Two examples of `wmqra_classification_info` elements follow:

```
<wmqra_classification_info transaction_class="TC_3"
  selector="JMSPriority>3 AND JMSPriority<8"
  destination="queue://QMGR1/Q1"
  queue_manager="QMGR1"
  description="medium priorities with a queue manager name of QMGR1 and
    a queue name of Q1 map to TC_3"/>
```

```
<wmqra_classification_info transaction_class="TC_4"
  destination="topic://a/b/*"
  description="Any topics starting with a/b/ map to TC_4"/>
```

selector

Use the `selector` attribute of the `wmqra_classification_info` element to assign a transaction class to a message based on its properties. This attribute can also be used to assign a transaction class to a message-driven bean that has a selector clause in its deployment descriptor:

- The value of the `selector` attribute is an SQL expression that selects a message according to the values of the message properties. The syntax is that of a message selector in the JMS 1.1 specification.
- The value of the `selector` attribute must have the correct syntax for an XML document. You must replace the `<` and `>` symbols with the entity references `<` and `>`, respectively. Similarly, if you use an apostrophe or quotation mark, use the `'` and `"` entity references.

destination

A URI representing the WebSphere MQ destination to which the message has been delivered. The classification applies to the destination named by this property, or to any destination if you do not specify this property. If the URI represents a queue type destination it can optionally include a queue manager name that matches the `queue_manager` attribute of the `wmqra_classification_info` element, if it is specified. If the URI represents a topic type destination, it can make use of wild cards. For more information on wild card support with WebSphere MQ see the WebSphere MQ information center.

queue_manager

The name of the WebSphere MQ queue manager to which the message has been delivered. The

classification applies to the queue manager named by this property, or to any queue manager if you do not specify this property. If the URI specified by the destination attribute contains a queue manager part, it must match this value. The queue manager name must conform to WebSphere MQ naming conventions.

description

Although the description field is optional, you must use it on all the `wmqra_classification_info` elements.

Each `wmqra_classification_info` element can contain one or more of these properties as needed to classify the work for a message. A `wmqra_classification_info` element cannot contain more than one instance of each property.

If a message matches several `wmqra_classification_info` elements, the element that appears first is used. For example, consider the following specifications:

```
<wmqra_classification_info queue_manager="QMGR1" transaction_class="TC_1" />
<wmqra_classification_info destination="queue:///Q1" transaction_class="TC_2" />
```

A message that arrives at destination Q1 on queue manager QMGR1 is assigned the classification "TC_1". A message that arrives at Q1 from another queue manager is assigned the classification "TC_2".

If a message does not match any `wmqra_classification_info` element in an enclosing `WMQRAClassification` element, the message is assigned the default classification from the `WMQRAClassification` element. If there are multiple `WMQRAClassification` elements, the default transaction class from the first `WMQRAClassification` element is used.

If no `WMQRAClassification` elements are defined, all work receives the default classification "WMQRA". You must perform z/OS Workload Manager actions that are required to use the TCLASS value "WMQRA", as described in "Classifying z/OS workload" on page 303.

Using transaction classes to classify workload for WLM

You can use transaction classes to classify client workload for workload management (WLM). The workload that WLM manages consists of different transactions that are targeted to separate servants, each with goals defined by specific service classes. The service classes chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive. You do not want to set a servant higher in the service class hierarchy than more important work such as production WebSphere, CICS, or IMS transaction servers.

Before you begin

Note: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests in a z/OS environment.

You must define the service objectives (goals) for your service classes. You must also define the service objectives of your servers. For more information about defining service objectives (goals) for each service class, see the *z/OS MVS Planning: Workload Management* book, SA22-7602, for example at <http://publibz.boulder.ibm.com/epubs/pdf/iea2w131.pdf>, or the z/OS WLM Web page at <http://www.ibm.com/servers/eserver/zseries/zos/wlm/>.

Note: You do not have to define special classification rules and work qualifiers initially. However, they should be defined before this system becomes a production system.

About this task

Each transaction is dispatched in its own WLM enclave in a servant process, and is managed according to the goals of its service class. The service class chosen also determines the WLM goal when Java Garbage Collection (GC) is running, which can be CPU intensive.

You should classify the servants to a high STC importance service class so that they are initialized quickly when WLM determines they are needed. However, you do not want to set a servant higher in the service class hierarchy than more important work such as CICS, or IMS transaction servers.

Controllers perform some processing as they receive work into the system, manage the transport handlers, classify a work item, and handle housekeeping tasks. Therefore, controllers should also be classified a high STC importance service class.

You can use the WLM CB-type classification criteria to classify work items:

- Server name (CN)
- Server instance name (SI)
- User ID assigned to the transaction (UI)
- Transaction class (TC)

To classify work using server and userid criteria, use a combination of the WLM Workload Classification rules in the WLM ISPF dialog panels. For more information about defining WLM Classification rules, see *Workload management (WLM)* and its related article that includes an example of classification rules.

To classify work using transaction classes, you define and use transaction class mappings, as described in this task. The following steps are used to classify work using transaction classes:

1. Define transaction class mappings based on the HTTP virtual host name, port number, and URI (Universal Resource Identifier - encoded address for any resource on the Web) provided with each work HTTP or HTTPS request.
 - a. Create a Transaction Class mapping file (as a simple text file). For example: `/wasconfig/t5was/MyTrMapFile.txt`

Note: This file must be in EBCDIC format.

- b. Edit the Transaction Class mapping file to define each transaction class mapping that you want to use. Define each mapping on a separate line, using the following syntax:

```
TransClassMap host:port uritemplate tclass
```

Note: You can use wildcard characters for the host and port fields if you use them for both fields. For example:

```
TransClassMap wsc4.washington.ibm.com:9080 /MyIVT/index.* TCLMYIVT
TransClassMap wsc4.washington.ibm.com:9080 /MyIVT/ivtejb TCLMYEJB
TransClassMap wsc4.washington.ibm.com:* /SuperSnoop* TCLSNOOP
TransClassMap wsc4.washington.ibm.com:* /ssb/* TCLSSB
TransClassMap *:* /admin* TCLADMIN
```

2. Specify the Transaction Class mapping file on the administrative properties for each server that is to handle work classified by transaction class. To specify the Transaction Class mapping file for a server:
 - a. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***. Then, in the Container Settings section, click **Container settings > Web container**.
 - b. In the Additional Properties section, click **z/OS additional settings**.
 - c. In the **Transaction Class Mapping** field, the fully qualified name of the Transaction Class mapping file that you edited in a previous step. For example: `/wasconfig/t5was/MyTrMapFile.txt`

- d. If you want to use a transaction class to classify outbound data that is delivered in response to HTTP and HTTPS requests, select the TCLASS option in the **Network QoS** field. If you specify TCLASS, the product uses the transaction class value that was used to classify the inbound request to the z/OS Workload Manager.

Example

The following table shows classification rules for STC-type work that covers the controller and servant regions started tasks:

Action	-----Qualifier-----			-----Class-----	
	Type	Name	Start	Service	Report
				DEFAULTS:	OPS_DEF
_____	1	TN	%DMN	_____	OPS_HIGH
_____	1	TN	T5SRV*	_____	RWSDMN
_____	1	TN	WS%%%	_____	OPS_MED
_____	1	TN	WS%%S	_____	RT5SRV
					SYSSCT
					RWSCCLR
					OPS_HIGH
					RWSSRVR

The following table shows classification rules for CB-type work in which the default service class is WSMED and has a reporting class of RWSDEFLT. Work run in the WSPROD server is classified as WSMED with a reporting class of RWSPROD, unless it has a transaction class of TCLASS1, TCLASS2, or TCLASS2 assigned through the transaction class mapping file below.

Qualifier #	Qualifier type	Qualifier name	Start position	Service Class	Report Class
				Default:	WSMED
					RWSDEFLT
1	CN	WSPROD	1	WSMED	RWSPROD
2 .	TC	. TCLASS1		WSFAST	RWSPRD1
2 .	TC	. TCLASS2		WSMED	RWSPRD2
2 .	TC	. TCLASS5		WSSLOW	RWSPRD5
1	CN	WSTEST	1	WSSLOW	RTSTEST
2 .	UI	. USER1		WSMED	RTSTSTU2
2 .	TC	. TCLASS5		WSSLOW	RTSTST5

The following table shows how work can be assigned a transaction class based on its host name, port number, or URI. For example, a web request of `http://ibm.com:80/Webap1/myservlet` handled by the WSPROD server would be assigned a transaction class of TCLASS1, a service class of WSFAST, and a reporting class of RWSPRD1 by the classification rules shown above.

```
TransClassMap www.ibm.com:80 /Webap1/myservlet TCLASS1
TransClassMap www.ibm.com:* /Webap1/myservlet TCLASS2
TransClassMap *:443 * TCLASS3
TransClassMap ** /Webap1/myservlet TCLASS4
TransClassMap www.ibm.com:* /Webap5/* TCLASS5
TransClassMap * * TCLASS6
```

Transaction class mapping file entries

Transaction class mapping file entries indicate the workload management (WLM) goal for each class of client work. Each client transaction is dispatched in its own WLM enclave in a servant region process, and is managed according to the goals specified for its service class.

Note: Transaction class mapping file support is deprecated. You should use a workload classification document instead of a transaction class mapping file to classify work requests.

Following is the syntax for entries in a transaction class mapping file:

```
TransClassMap host:port uritemplate tclass
```

where:

host Is the value compared against the hostname of the HOST: header of the request.

Note: You cannot use wild-card characters in the host field unless you use it for the entire field; for example `*:*`.

port Is the value compared against the port of the request.

Note: You cannot use wild-card characters in the `or` port field unless you use it for the entire field; for example `*:*`.

uritemplate

Is the value compared against the URI of the request. Any query string will not be used in the comparison. This value can be a wildcard `**`, or end in a wildcard.

tclass Is the Workload Manager Transaction Class name that will be used in the creation of the enclave.

Examples:

```
TransClassMap www.ibm.com:80 /webap1/myservlet TCLASS1
```

```
TransClassMap www.ibm.com:* /webap1/myservlet TCLASS2
```

```
TransClassMap *:443 * TCLASS3
```

```
TransClassMap *:*/webap1/myservlet TCLASS4
```

```
TransClassMap www.ibm.com:* /webap2/* TCLASS5
```

```
TransClassMap * /myservlet TCLASS6
```

```
TransClassMap * * TCLASS6
```

Controller and Servant WLM classifications

Applications are deployed on a server or a cluster of servers. Each server consists of a controller and one or more servants. Each controller is started by the deployment manager, or by an MVS operator command as an MVS started tasks. Each servant is started by the Workload manager (WLM) as it is needed.

You can use transaction classes to classify client workload for WLM. Transaction classification can be based on the following WLM classification criteria:

- Server name (CN) – if the server is clustered, this value is the short name for the cluster; if the server is not clustered, this value is the value specified on the server custom property, `ClusterTransitionName`.
- The Server instance name (SI) – this is the short name for the server. This is usually not very useful because you cannot control which server instance runs a transaction within a cluster.
- User ID assigned to the transaction (UI) Transaction class (TC) - use the transaction class mapping file for the server to assign this ID to a transaction

.

Controller classification

You should classify controllers in `SYSSTC` or give them a high importance and velocity goal, because controllers do some of the processing that is required to receive work into the system, manage the HTTP transport handler, classify the work, and do other housekeeping tasks,

Note: A step in controller start-up procedures, such as `BBO7ACR`, invoke the `BPXBATCH` program to check the service levels delivered, applied, and pending, and log the results in the `/properties/service/logs/applyPTF.log` file. Because the `BPXBATCH` program is classified according the OMVS rules, instead of inheriting the service classification of the startup procedure, on a busy system several minutes might pass before `BPXBATA2` gets control. You can minimize the impact of the `BPXBATCH` step by changing the WLM Workload Classification Rules for OMVS work to a higher service objective. In the following example, OMVS work is assigned a `EBIZ_HI` service class, which has an importance of 1, and Velocity of 50.

```

Subsystem Type . : OMVS
Description . . . E_Biz Classification Rule
-----Qualifier----- -----Class-----
Action Type Name Start Service Report
DEFAULTS: EBIZ_DEF _____
_____ 1 TN FTPSERVE _____ EBIZ_HI _____
_____ 1 UI OMVSKERN _____ SYSSTC _____
_____ 1 TN WSSRV* _____ EBIZ_HI RPTACR <<==

```

Servant classification

When a servant region starts, WLM classifies the servant region as a started task, and places it in a service class and a report class based on the classification rules that are specified in the WLM policy. WLM uses this service class to determine the priority of the address space. The priority of an address space determines its access to system resources during the initialization of the servant region.

Each work request that the controller region receives is associated with a WLM enclave. Each of these enclaves is then associated with a WLM service class and report class. When WLM distributes the work requests from the controller region to the various servant regions, it tries to keep all of the work requests that are associated with the same service class together. This grouping of requests means that the work requests processed by a given servant region are associated with enclaves that are usually associated with the same service class. If all of the enclaves are associated with the same service class, it can be said that the servant region is associated with that service class.

After the servant region is initialized and starts to receive work requests, all of the dispatch threads and non-dispatch threads, such as the Java Garbage Collection (GC) threads, are managed according to the goals that are associated with the WLM service class of the enclaves associated with the work requests that are executing in the servant.

All of the resources that the dispatch threads consume while an enclave is associated with those threads are reported in the report class associated with the enclave. All of the resources that the threads that are not associated with an enclave consume are reported in the report class associated with servant region started task.

Note: You should make sure that the servant classification is not higher in the service class hierarchy than more important work, such as the controller and CICS, or IMS transaction servers.

WLM Classification Rules for STC-type work

Here is a simple example of the WLM Classification Rules for STC-type work that covers the controller and servant started tasks:

Action	Type	Name	Start	Service	Report	
				DEFAULTS: OPS_DEF		
_____	1 TN	%%DMN	_____	OPS_HIGH	RWSDMN	_____ 1 TN T5SRV* _____ OPS_MED
_____	1 TN	WS%%%	_____	SYSSTC	RWCTLR	
_____	1 TN	WS%%%S	_____	OPS_HIGH	RWSSRV	

Creating clusters

A cluster is a set of application servers that you manage together as a way to balance workload.

Before you begin

Before you create a cluster:

- Review the content of the topic "Clusters and workload management," especially the information about setting cluster weights.

- Decide if you want enterprise bean requests routed to the node on which the client resides.
- Decide if you want to use HTTP memory-to-memory replication.
- Determine the appropriate configuration settings for the first cluster member. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.
- Decide on which node you want the first cluster member to reside.

About this task

You might want to create a cluster if you need to:

- Balance your client requests across multiple application servers.
- Provide a highly available environment for your applications.

A cluster enables you to manage a group of application servers as a single unit, and distribute client requests among the application servers that are members of the cluster.

If you plan to create a cluster of servers that spans multiple systems in a sysplex and has stateful session beans with an activation policy of Transaction deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers are running.

To create a cluster:

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**. The Create a new cluster wizard starts.
2. Specify a name for the cluster.
3. Optional: Specify a short name for the cluster.

For clustered servers, the WLM application environment is the default value for the cluster short name. If you specify a short name for a cluster, the name:

- Must be one to eight characters in length.
- Must contain only alpha-numeric or national language characters.
- Cannot start with a number.
- Must be unique in the cell.
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

Note: If you specify a short name, make sure that you set up a RACF SERVER class profile that includes this short name.

4. Select **Prefer local** if you want to enable host-scoped routing optimization. This option is enabled by default. When this option is enabled, if possible, EJB requests are routed to the client host. This option improves performance because client requests are sent to local enterprise beans.
5. Select **Configure HTTP session memory-to-memory replication** if you want a memory-to-memory replication domain created for this cluster. The replication domain is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled. Also, the Web container for each cluster member is configured for memory-to-memory replication.

To change these settings for the replication domain, click **Environment > Replication domains > *replication_domain_name***. To modify the Web container settings, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members > *cluster_member_name***. Then, in the Container settings section, click **Web container settings > Web container > Session management > Distributed environment settings** in the administrative console. If you change these settings for one cluster member, you might also need to change them for the other members of this cluster.

6. Click **Next**.

7. Choose whether to create an empty cluster or to create the first member of the cluster.

If you decide to create an empty cluster, to add members to this cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > cluster_name > Clusters members > New**.

To create an empty cluster:

- a. Select **None. Create an empty cluster**.
- b. Click **Next** to display a summary of the defined cluster.
- c. Click **Finish** to create the cluster, or click **Cancel** if you decide not to create this cluster.

When you create the first cluster member, remember that a copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

- a. Specify the name of the first cluster member.
- b. Select the node on which you want this cluster member to reside.
- c. Specify a short name for this cluster member. The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, and RACF.
- d. Specify the weight value for the cluster member. The weight value controls the amount of work that is directed to the application server. If the weight value for this server is greater than the weight values that are assigned to other servers in the cluster, then this server receives a larger share of the workload. The weight value represents a relative proportion of the workload that is assigned to a particular application server. The value can range from 0 to 20.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
 - For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
 - Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.
- e. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server. When this option is selected, which is the default setting, this cluster member does not have HTTP transports or HTTP transport channels that conflict with any of the other servers that are defined on the same node. If you unselect this option, all of the cluster members will share the same HTTP ports.
 - f. Select the core group to which you want this cluster member to belong. You are prompted for the core group only if you have more than one core group defined for this cluster.
 - g. Select one of the following options as the basis for the first cluster member.
 - Create the member using an application server template.

If you select the **defaultZOS** template, which is the only one that is listed unless you used the **createServerTemplate** command for the AdminTask object to create additional templates, the first cluster member uses the default port assignments for z/OS. If some of these ports are already defined for use elsewhere in your system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.
 - Create the member using an existing application server as a template.
 - Create the member by converting an existing application server.

Note: You can only add an existing application server to the cluster if you select that server as the first cluster member. You cannot add other existing application servers to that cluster after you create the first cluster member. If you add an existing server to a cluster, the only way to remove that server from the cluster is to delete the server. Therefore, you might want to use the existing server as a template for the first cluster member instead of as the cluster member. If you keep the original application server out of the cluster, you can reuse that server as the template if you need to rebuild the configuration.

8. Click **Next**.
9. Create additional cluster members. Before you create additional cluster members, check the configuration settings of the first cluster member. These settings are displayed at the bottom of the Create additional cluster members panel of the Create a new cluster wizard. For each additional member that you want to create:
 - a. Specify a unique name for the member. The name must be unique within the node.
 - b. Select the node to which you want to assign the cluster member.
 - c. Specify the weight you want given to this member. The weight value controls the amount of work that is directed to the application server. If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, then the server receives a larger share of the workload. The value can range from 0 to 20.
 - d. Specify a short name for this cluster member. The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, and RACF.
 - e. Select **Generate unique HTTP ports** if you want to generate unique port numbers for every HTTP transport that is defined in the source server.
 - f. Click **Add member**. You can edit the configuration settings of any of the newly created cluster members other than the first cluster member, or you can create additional cluster members. Click **Previous** to edit the properties of the first cluster member. The settings for the first cluster member become the settings for the cluster member template that is automatically created when you create the first cluster member.
10. When you finish creating cluster members, click **Next**.
11. View the summary of the cluster and then click **Finish** to create the cluster, click **Previous** to return to the previous wizard panel and change the cluster, or click **Cancel** to exit the wizard without creating the cluster.
12. To further configure a cluster, click **Servers > Clusters > WebSphere application server clusters >** , and then click the name of the cluster. Only the **Configuration** and **Local Topology** tabs appear until you save your changes.
13. Click **Review** to review your cluster configuration settings. Repeat the previous step if you need to make additional configuration changes.
14. If you do not want to make any additional configuration changes, select Synchronize changes with Nodes and then click **Save**. Your changes are saved and synchronized across all of your nodes.

Note: If you click **Save**, but do not select Synchronize changes with Nodes, when you restart the cluster, the product does not start the cluster servers because it cannot find them on the node. If you want to always synchronize your configuration changes across your nodes, you can select Synchronize changes with Nodes as one of your console preferences.

15. Restart the cluster.

Results

You have created a cluster to which you can assign work requests. The **Runtime** and **Local Topology** tabs appear the next time you access this page.

What to do next

- You can click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members** in the administrative console, and then click the name of a cluster member to view all of the configuration settings for this cluster member. You can then use this page to change some of the configuration settings for the selected cluster member.

For example, if you do not need to have all of the cluster member components start during the cluster startup process, you might want to reconfigure the cluster members, such that the **Start components as needed** is selected. This option is not selected when a new cluster member is created. Selecting this option can improve cluster startup time, and reduce the memory footprint of the cluster members.

Note: Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

Note: The default addressing mode for a new server is 64-bit. You can deselect the Run in 64-bit mode field if you need to use 31-bit addressing mode. However, support for running a server in 31-bit mode is deprecated.

- Use the administrative console to view or change the configuration settings for a cluster. For example, if you are running in a high availability environment, you can click **Servers > Clusters > WebSphere application server clusters > *cluster_name***, and then select the **Enable failover of transaction log recovery** option for this cluster. This option allows the recovery of transactions to failover from one cluster member to another.
- If you create the cluster member by converting an existing application server and that server is a member of a bus, you need to migrate the messaging engine in the server to the scope of a cluster. To do this, use the wsadmin command migrateServerMEtoCluster. Do not delete the messaging engine at server scope and re-create it a cluster scope, because this would prevent the messaging engine from working with previously configured destinations.
- Create additional cluster members.
- Start the cluster.
- Use scripting to automate the task of creating clusters.
- Create a static routing table to temporarily handle IIOIP routing for the cluster if your high availability infrastructure is disabled.

Creating a cluster: Basic cluster settings

Use this page to enter the basic settings for a cluster.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters > New**.

Cluster name

Specifies the name of the cluster. The cluster name must be unique within the cell.

Short name

Specifies the short name for this cluster. This field only appears if you are running on z/OS.

The short name is used as the WLM APPLENV name for all servers that are part of this cluster.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Configure HTTP session memory-to-memory replication

Specifies that when the cluster is created, a memory-to-memory replication domain is created for each of the members of this cluster.

If a replication domain is created, it is given the same name as the cluster and is configured with the default settings for a replication domain. When the default settings are in effect, a single replica is created for each piece of data and encryption is disabled.

To modify the replication domain settings, in the administrative console, click **Environment > Replication domains > *replication_domain_name***.

The default mode setting for the replication domain is Both `client` and `server`. In this mode, all data sent to either the client or the server is replicated. This setting is good for an environment that has a middle to low traffic load. However, if your environment has a high traffic load, you should change the replication domain mode setting to either `Client only`, or `Server only`, because these settings provide better scaling. In `Client only` mode, only data sent to the client is replicated. In `Server only` mode, only data sent to the server is replicated.

To modify the mode setting for a replication domain:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***.
2. Under Container settings, click either **SIP container settings** or **Web container settings**, and then click **Session management > Distributed environment settings > Memory-to-memory replication**.
3. Select a different mode. It does not matter whether you change the mode under the SIP container or the Web container because the same replication domain settings apply to both containers.

Note: If you change any of the replication domain settings for one cluster member, including the mode setting, you should change them for all of the other members of the cluster.

Creating a cluster: Create first cluster member

Use this page to specify settings for the first cluster member.

There are two ways to create the first member of a cluster:

- You can create the first member when you create a new cluster.
To create a new cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**.
- You can create an empty cluster and then add a first member after you finish creating the cluster.
To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members > New**.

When you create the first cluster member, a copy of that member is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

When adding servers to a cluster, remember that the only way to remove an application server from a cluster is to delete the application server from the list of cluster members.

Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name

Specifies the short name for this cluster member. This field only applies to the z/OS platform.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, WebSphere Application Server for z/OS assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number.
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight

Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no effect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Core group

Specifies the core group in which the application server resides. This field displays only if you have multiple core groups configured. You can change this value only for the first cluster member.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Select basis for first cluster member:

Specifies the basis you want to use for the first cluster member.

- If you select **Create the member using an application server template**, the settings for the new application server are identical to the settings of the application server template you select from the list of available templates.

If you select the defaultZOS template, which is the only one that is listed unless you used the `createServerTemplate` command for the AdminTask object to create additional templates, the first cluster member uses the default port assignments for the z/OS platform. If some of these ports are

already defined for use elsewhere in your z/OS system, your newly created cluster member might not start, might function incorrectly, or might generate unexpected error messages. Therefore, you must resolve any port conflicts before you start this server.

- If you select **Create the member using an existing application server as a template**, the settings for the new application server are identical to the settings of the application server you select from the list of existing application servers.
- If you select **Create the member by converting an existing application server**, the application server you select from the list of available application servers becomes a member of this cluster.
- If you select **None. Create an empty cluster**, a new cluster is created but it does not contain any cluster members.

Note: The basis options are available only for the first cluster member. All other members of a cluster are based on the cluster member template which is created from the first cluster member.

Creating a cluster: Summary settings

Use this administrative console page to view and save settings when you create a cluster or cluster member.

You can view this administrative console page whenever you create a new cluster or a new cluster member. This summary page displays your configuration changes before you commit the changes and the new cluster or cluster member is created.

To create a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > New**.

To create a cluster member for an existing cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members > New**

The bounding node group of the cluster is based on the first application server that is added as a member of the cluster. The settings for this first member become the settings for the cluster member template that is then used to create additional cluster members. To select a different bounding node group, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members > New**, and then select the appropriate node group for the new cluster member. When you select a different node group, another cluster member template is automatically created for that node group, if one does not already exist.

Review the changes to your configuration, and then click **Finish** to complete and save your work.

Creating a cluster: Create additional cluster members

Use this page to create additional members for a cluster. You can add a member to a cluster when you create the cluster or after you create the cluster. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

To add members to a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members > New**. After you enter the required information about the new cluster member, click **Add Member** to add this member to the cluster member list.

After adding a cluster member, you might need to change one or more of the property settings for this cluster member, or another cluster member that you just added. To change one or more property settings for any cluster member that you just added, other than the first cluster member, select that cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

If you decide not to create a particular cluster member, select the member and then click **Delete**.

You cannot edit or delete the first cluster member or an already existing cluster member.

If you create additional cluster members immediately after you create the first cluster member, the list of cluster members includes a checklist in front of the names of these additional cluster members. However, a check box does not appear in front of the name of the first cluster member because you cannot delete this member or edit its settings. To modify the first cluster member, click **Previous**.

Similarly, if you are adding cluster members to a cluster that already has existing members, the existing members appear in the list of cluster members but a check box does not appear in front of the names of these cluster members. To delete one of these existing members or to change the settings of one of these cluster members, in the administrative console click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members**, and then select the member that you want to delete or whose configuration settings you want to change.

Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the application server resides.

Short name

Specifies the short name for this cluster member. This field only displays if you are running on z/OS.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, the product assumes that you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Weight

Specifies the amount of work that is directed to the application server.

If the weight value for the server is greater than the weight values that are assigned to other servers in the cluster, the server receives a larger share of the cluster workload. The value can range from 0 to 20. Enter zero to indicate that you do not want requests to route to this application server unless this server is the only server that is available to receive requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the application server. By generating unique HTTP ports for the application server, you avoid potential port collisions and configurations that are not valid.

Server cluster collection

Use this page to view information about and change configuration settings for a cluster. A cluster consists of a group of application servers. If one of the application servers fails, requests are routed to other members of the cluster.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters**.

To define a new cluster, click **New** to start the Create a new cluster wizard.





Name

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Status

This field indicates whether the cluster is partially started, started, partially stopped, stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

After you click **Start** or **Ripplestart** to start a cluster, each server that is a member of that cluster launches if it is not already running. When the first member launches, the state changes to *partially started*. The state remains *partially started* until all cluster members are running. When all cluster members are running, the state changes to *running* and the status changes to *started*. Similarly, when you click **Stop** or **ImmediateStop** to stop a cluster, the state changes to *partially stopped* when the first member stops, and then changes to *stopped* when all cluster members are not running.

	Started	The server is running.
	Partially stopped	The server is in the process of changing from a started state to a stopped state.
	Stopped	The server is not running.
	Unknown	The server status cannot be determined.

Server cluster settings

Use this page to view or change the configuration of a server cluster instance, and to view the local topology of a server cluster instance.

To change the configuration and local topology of a server cluster, in the administrative console click **Servers > Clusters > Clusters > *cluster_name***.

To view runtime information, such as the state of the server cluster, click **Servers > Clusters > WebSphere application server clusters > cluster_name**, and then click the **Runtime** tab.

To display the topology of a specific cluster, click **Servers > Clusters > WebSphere application server clusters > cluster_name**, and then click the **Local Topology** tab.

If the high availability infrastructure is disabled and you require IIOIP routing capabilities, follow the instructions contained in the "Enabling static routing for a cluster" topic to create a static route table. This table enables the cluster to handle IIOIP requests.

Note:

- Because the information contained in the static route table does not account for server runtime state, you should only use this option if the high availability infrastructure is disabled.
Use of a static route table preempts the use of the dynamic routing table that is contained in cluster members. After the static file is transferred to a node, whenever a cluster member residing in that node starts, that cluster member uses the static table instead of the dynamic table to handle IIOIP routing. If a cluster member is running when you create the static route table, then you must restart that cluster member to give that cluster member access to the static route table information, because the table content is loaded at run time.
- After the table is created, an informational message, similar to the following message, is issued that indicates the name of the file that contains the table and where that file is located:

```
The route table for cluster MyCluster was exported to file
/home/myInstall/was/server/profiles/dmgrProfile/config/cells/
MyCell/clusters/Myfile.wsrttbl.
```

As this message indicates, the file containing the static route table is placed in the config directory of the deployment manager for this cluster. Keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.
- If you set up a static route table, then you must statically set the ORB_LISTENER_ADDRESS port on each of the cluster members because the route table is static, and the cluster members do not communicate during state changes. If this port is not assigned, then the cluster members restart on different ports, and the static routing information is not able to route requests to the cluster members.

Cluster name:

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

Short name:

Specifies the short name for this cluster. This field displays only if you are running on z/OS.

The short name is used as the WLM APPLENV name for all servers that are part of this cluster.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length
- Must contain only uppercase alphanumeric characters
- Cannot start with a number
- Must be unique in the cell

If you do not specify a short name, then the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Unique Id:

Specifies the unique ID of this cluster.

The unique ID property is read only. The system automatically generates the value.

Bounding node group name:

Specifies the node group that forms the boundaries for this cluster. All application servers that are members of a cluster must be on nodes that are members of the same node group.

A node group is a collection of application server nodes. A node is a logical grouping of managed servers, usually on a system that has a distinct IP host address. All application servers that are members of a cluster must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that clusters formed across the nodes in the node group can host the same application in each cluster member. A node must be a member of at least one node group and can be a member of more than one node group.

Create and manage node groups by clicking **System administration > Node groups** in the administrative console.

Enable failover of transaction log recovery:

Specifies that for the transaction service component, failover of the transaction log for recovery purposes is enabled or disabled. The default is disabled.

When this setting is enabled, and the transaction service properties required for peer recovery of failed application servers in a cluster are properly configured, failover recovery of the transaction log occurs if the server processing the transaction log fails. If the transaction services properties required for peer recovery of failed application servers in a cluster are not properly configured, then this setting is ignored.

State:

Specifies whether the cluster is stopped, starting, or running.

If all cluster members are stopped, the cluster state is stopped. After you request to start a cluster, the cluster state briefly changes to starting and each server that is a member of that cluster launches, if it is not already running. When the first member launches, the state changes to `websphere.cluster.partial.start`. The state remains partially started until all cluster members are running, then the state changes to running. Similarly, when stopping a cluster, the state changes to partially stopped as the first member stops and changes to stopped when all members are not running.

Valid values

starting, partially started, running, partially stopped, or stopped.

Cluster topology

Use this page to display, in a tree format, a list of all of the application server clusters defined for your WebSphere Application Server environment. The list shows all of the nodes and cluster members that are included in each cluster contained in a cell.

To view this page, in the administrative console, click **Servers > Clusters > Cluster topology**.

Enabling static routing for a cluster

If your high availability infrastructure is disabled and you require IOP routing capabilities, you can create a static routing table for the members of a cluster to use to handle enterprise bean requests. Because the

information contained in this static routing table does not account for server runtime state, you should delete this table and return to using the dynamic routing table as soon as your high availability infrastructure is enabled.

Before you begin

Before you create a static route table, ensure that:

- The ORB_LISTENER_ADDRESS port is set to a non-zero value on each of the cluster members. Because the route table you create is static, and the cluster members do not communicate during state changes, if you do not set the ORB_LISTENER_ADDRESS port on each of the cluster members, the cluster members might restart on different ports, and IIOp requests will not be routed correctly.

To change the value specified for the ORB_LISTENER_ADDRESS port:

1. In the administrative console, click **Servers > Server Types > WebSphere application servers > *server_name***, and then under Communications, click **Ports**.
 2. Click **ORB_LISTENER_ADDRESS** in the Port name field.
 3. Change the value specified for the Port field to a value that is greater than 0.
- Each cluster member is started and can use these new non-zero ORB_LISTENER_ADDRESS port values to correctly route IIOp requests.

About this task

You should only create a static route table if your high availability infrastructure is disabled and you require IIOp routing capabilities. To create a static route table:

1. Start the wsadmin tool if it is not already running.
2. Identify the cluster managed bean (MBean) for the cluster for which you are creating the route table, and assign that MBean to a variable.

- Using Jacl:

```
cluster = AdminControl.completeObjectName('cell=
cell_name,type=Cluster,name=cluster_name,*)
print cluster
```

- Using Jython:

```
set cluster [AdminControl completeObjectName cell=
cell_name,type=Cluster,name=cluster_name,*]
puts $cluster
```

These commands return the name of the cluster MBean for the specified cluster. For example, for cluster cluster1, the output from these commands will be similar to the following message:

```
WebSphere:cell=mycell,name=cluster1,mbeanIdentifier=Cluster,type=
Cluster,process=cluster1
```

3. Export the route table.

- Using Jacl:

```
$AdminControl invoke $cluster exportRouteTable
```

- Using Jython:

```
AdminControl.invoke(cluster, 'exportRouteTable')
```

After the table is created, the name of the route table file, is displayed in a message similar to the following message:

```
/home/myInstall/was/server/profiles/dmgrProfile/config/cells/mycell/
clusters/cluster1/cluster1.wsrctl
```

As this message illustrates, the file containing the table is placed in the config directory of the deployment manager for that cluster. You should keep a record of this location so that you can delete this file when you are ready to start using dynamic routing again.

4. Synchronize the configuration changes across nodes.

- a. Clear the configuration repository Epoch. If you do not clear the configuration repository Epoch, the synchronization only updates the files that the configure service component edited, which does not include the file that contains the static routing table.

Using Jacl:

```
set configRepository [$AdminControl completeObjectName  
    node=node_name,type=ConfigRepository,*]  
$AdminControl invoke $configRepository refreshRepositoryEpoch
```

Using Jython:

```
configRepository = AdminControl.completeObjectName('node=node_name,  
    type=ConfigRepository,*')  
AdminControl.invoke(configRepository, 'refreshRepositoryEpoch')
```

- b. Repeat this process for each node that you want to synchronize.
5. Stop the cluster. Follow the instructions specified either the *Stopping clusters* or *Stopping clusters using scripting* topic.
6. Exit the wsadmin tool.
7. Use the following Debug flag appended to the startServer command to manually start each member of this cluster.

```
-Dcom.ibm.websphere.management.registerServerIORWithLSD=false
```

For example, to start server1 on a Windows operating system with static routing enabled, issue the following command from the server profile's bin directory:

```
startServer.bat server1 -Dcom.ibm.websphere.management.registerServerIORWithLSD=false
```

Results

The cluster members use the static route table to perform IIOF routes.

What to do next

When your high availability infrastructure is enabled, follow the instructions in the topic *Disabling static routing for a cluster* to disable static routing. When static routing is disabled, the cluster members resume using dynamic routing.

Disabling static routing for a cluster

Because the information contained in a static routing table does not account for server runtime state, you should delete this table and return to using the dynamic routing table as soon as your high availability infrastructure is enabled. When you delete the static routing table, cluster members automatically resume using dynamic routing to handle enterprise bean requests.

About this task

Perform the following steps to delete the static routing table.

1. For each member of the cluster, set the ORB_LISTENER_ADDRESS port to 0 (zero).
 - a. In the administrative console, click **Servers > Server Types > WebSphere application servers > server_name**, and then in the Communications section, click **Ports**.
 - b. Click **ORB_LISTENER_ADDRESS** in the Port name field.
 - c. Change the value specified for the Port field to 0.
2. Manually delete the static route table file from the config directory of the deployment manager for the cluster.

The path to this config directory was included in the message that you received when you originally exported this file. If you did not retain this information, you can do a search in the deployment manager config directory for the file cluster_name.wsrttbl.

3. Synchronize the configuration changes across nodes.

- a. Clear the configuration repository Epoch. If you do not clear the configuration repository Epoch, the synchronization only updates the files that the configure service component edited, which does not include the file that contains the static routing table.

Using Jacl:

```
set configRepository [$AdminControl completeObjectName  
    node=node_name,type=ConfigRepository,*]  
$AdminControl invoke $configRepository refreshRepositoryEpoch
```

Using Jython:

```
configRepository = AdminControl.completeObjectName('node=node_name,  
    type=ConfigRepository,*')  
AdminControl.invoke(configRepository, 'refreshRepositoryEpoch')
```

- b. Repeat this process for each node that you want to synchronize.
4. Stop the cluster. Follow the instructions specified either the *Stopping clusters* or *Stopping clusters using scripting* topic.
5. Start the cluster again. Follow the instructions specified either the *Starting clusters* or *Starting clusters using scripting* topic.
6. Exit the wsadmin tool.

Results

The cluster members resume using the dynamic routing table to handle IIOp requests.

Clusters on which stateful session beans will be deployed

For a cluster of servers that span multiple systems in a sysplex, and that will have stateful session beans with an activation policy of *Transaction* deployed in them, the passivation directory should reside on an HFS (hierarchical file system) that is shared across the multiple systems in the sysplex on which the clustered servers will be running.

Before creating the cluster, it is important to consider the following:

- Does the cluster have cluster members that are on different systems in the sysplex?
- If so, do any of the applications that are to be deployed or are already deployed have stateful session beans?
- If so, do the Stateful Session beans have an activation policy of *Transaction*?

If you answered yes to all the questions above, then you should define a shared HFS (hierarchical file system) to be used as the passivation directory for the stateful session beans.

The name of the passivation directory should contain the install root and cluster name. In the following example, `/WebSphere/V6R0M0/AppServer` is also known as `USER_INSTALL_ROOT`, and the name of the cluster is **cluster1**.

```
/WebSphere/V6R0M0/AppServer/passivation/cluster1
```

For optimal performance, you should create a passivation directory for each cluster. Also, the default passivation directory should not be used for clusters; it should be used for non-clustered servers and servers that do not have stateful session beans with an activation policy of *Transaction*.

For information about defining a shared HFS, see *z/OS UNIX System Services Planning*.

For information on specifying the passivation directory with the administrative console, see the *Administering applications and their environment* PDF.

Adding members to a cluster

You can use clusters to balance workload in an environment containing multiple application servers.

Before you begin

Create a cluster if you do not already have a cluster defined for your environment.

About this task

If you are migrating from a previous version of the product, you can upgrade a portion of the nodes in a cell, while leaving others at the previous release level. For a time, you might be managing servers that are at a previous release level, and servers that are running at the current release level in the same cell.

When you create a cluster, you specify the node on which the first cluster member resides. In a mixed cell environment, you can use any server from within that node group to create a new cluster member. For example, if the node group to which the cluster belongs consists of a Version 7.0 node, and a Version 6.1 node, you can use a server from either the Version 6.1 or the Version 7.0 node to create a new cluster member.

Use the following procedure to create a new cluster member, view information about existing cluster members, or manage existing cluster members.

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members**. The Cluster members page lists members of a cluster, and for each member indicates:
 - The node on which the member resides.
 - The version of the application server. This information specifies whether the cluster is a mixed cluster.
 - The configured weight for the member.
 - The runtime weight for the member. This weight indicates the proportionate workload that is currently directed to this cluster member.
 - Whether the member is started, stopped, or encountering problems.
2. Click **New** to create a new cluster member.

Clicking **New** starts the Create a new cluster member wizard. Use this wizard to add new members to an already configured cluster.

A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create. Usually, only one template is available for you to use to create additional cluster members for a cluster. However, if a cluster includes nodes that are at different versions of the product, there is a different template for each version. For example, if a cluster has cluster members that reside on both a Version 6.1 node and a Version 7.0 node, the cluster has two templates. The Version 6.1 template is used when you create an additional cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional cluster member on the Version 7.0 node.

To view the cluster member templates that are available for creating a new member of a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members > Templates**.

- a. Specify a name for the application server that you are defining as a cluster member. The name must be unique within the node.
- b. Select the node for the cluster member.
- c. Specify a short name for this cluster member. The short name is the default job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, RACF, and started task control.

d. Specify the server weight.

The weight value you specify controls the number of requests that are directed to the application server. Even though you specify a value of 0 to 20 as the weight of a server, the weight that is given to the server as a member of a cluster is a proportion that is based on the weight assigned to the server, and the sum of the weights of all members of the cluster. In this proportion, the weight that is assigned to the server is the numerator, and the sum of the weights of all members of the cluster is the denominator.

When you add a new member to a cluster, the number of requests that are sent to each server in the cluster decreases, assuming that the number of requests coming into the cluster stays the same. Similarly when you remove a new member from a cluster, the number of requests that are sent to each server in the cluster increases, assuming that the number of requests coming into the cluster stays the same.

For example, if you have a cluster that consists of members A, B, and C with weights 2, 3, and 4, respectively, then 2/9 of the requests are assigned to member A, 3/9 are assigned to member B, and 4/9 are assigned to member C. If a new member, member D, is added to the cluster and member D has a weight of 5, then member A now gets 2/14 of the requests, member B gets 3/14 of the requests, member C gets 4/14 of the requests, and member D gets 5/14 of the requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.
- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no effect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

e. Specify whether to generate unique HTTP ports.

f. Click **Add member** to finish defining the cluster member. The first cluster member for this cluster is used as the template for this cluster member. You can repeat these steps to define other cluster members.

g. When you finish defining additional cluster members, review the summary information for the new cluster members. If you have to change any of the property settings for any of the new members, select that cluster member, and then click **Edit**. When you finish changing the property settings, click **Update member** to save your changes.

h. When you finish defining new cluster members, click **Next** to view the summary page for the cluster, and then click **Finish** to create these new cluster members.

3. Click **Review**, select **Synchronize changes with nodes**, and then click **Save** to save your changes.

Results

You created application servers that are members of an existing server cluster.

What to do next

If, when you created the new members, you chose to generate unique ports, update the alias list for the virtual host that you plan to use with the new servers.

You can also perform the following actions:

- On the Cluster members page in the administrative console, click the name of one of the cluster members, and examine the configuration settings for that cluster member. You can change any of the settings that are not appropriate.

For example, if you do not need to have all of the cluster member components start during the cluster member startup process, you might want to select **Start components as needed**, which is not automatically selected when a new cluster member is created. When this property is selected, cluster member components are dynamically started as they are needed. When this property is not selected, all of the cluster member components are started during the startup process. Therefore, selecting this property usually results in improved startup performance because fewer components are started during the startup process.

- Click **Servers > Sever Types > WebSphere application servers > *sever_name*** or click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members > *cluster_member_name*** to perform either of the following tasks:
 - Specify additional application server properties for this cluster member.
 - Click **Installed applications**.
- Start the cluster.
- Use scripting to automate the task of adding cluster members.

Cluster member collection

Use this page to view and manage application servers that belong to a cluster. You can also use this page to change the weight of any of the listed application servers.

Application servers that are part of a cluster are referred to as cluster members. A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

You can use this page to perform the following actions for the listed cluster members:

- Start a cluster member. To start a cluster member, select the server that you want to start. Then click **Start**.
- Restart a cluster member. To restart a cluster member, select the server that you want to restart. Then click **Restart**. When you restart a cluster member, the selected cluster member stops, following the normal server quiesce process, and then starts again.
- Stop a cluster member. To stop a cluster member, select the server that you want to stop, and then click one of the following buttons:

Stop When you click this button, the normal server quiesce process is followed. This process allows in-flight requests to complete before the entire server process shuts down.

Immediate Stop

When you click this button, the selected sever stops but the normal server quiesce process is not followed. This shutdown mode is faster than the normal server stop processing, but some application clients might receive exceptions if an in-flight request does not complete before the server process shuts down.

Terminate

You should only click **Terminate** if the cluster member does not respond when you click **Stop** or **Immediate Stop** or when you issue the Stop or ImmediateStop commands. Some application clients can receive exceptions. Therefore, you should always attempt an immediate stop before clicking **Terminate**.

Make Idle

When you click this button, the cluster member moves to the idle state.

- Delete a cluster member. To delete a cluster member, select the cluster member that you want to delete. Then click **Delete**.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You must use wsadmin commands to modify this template. Similarly, any changes that you make to the template do not affect existing cluster members.

See the *Using the administrative clients* PDF for more information on how to modify this template.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters > cluster_name > Clusters members**.

Member name

Specifies the name of the server in the cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the application server.

Node

Specifies the name of the node for the cluster member.

Host Name

Specifies the IP address, the full domain name system (DNS) host name with a domain name suffix, or the short DNS host name for the cluster member.

Version

Specifies the version of the product on which the cluster member runs.

Configured weight

Specifies the weight that is currently configured for the cluster member. The weight determines the amount of work that is directed to the cluster member. If the weight value for the server is greater than the weight values assigned to other servers in the cluster, the server receives a larger share of the cluster workload.

To change the configured weight for a cluster member you can either specify a new weight in the Configured weight field and click the **Update** button for the Configured weight column, or click on the name of the cluster member. Clicking the name of the cluster member navigates you to the page where you can change any of the configuration settings for that cluster member.

Runtime weight





Specifies the proportionate workload that is currently directed to the cluster member in comparison to the runtime weights of the rest of the cluster members. The runtime weight only applies for Remote Method Invocation over Internet Inter-ORB Protocol (RMI-IIOP) requests, Enterprise JavaBeans (EJB) requests, and HTTP requests received through a WebSphere proxy server.

You can use the **Runtime weight** property to dynamically adjust the weight that is given to a particular cluster member without having to stop and then restart that cluster member. To change the proportion, specify a new weight for the **Runtime weight** property, and then click the **Update** button for the Runtime weight column. The runtime weight change takes effect immediately.

Note: Changing the runtime weight for a cluster member does not affect the configured weight setting for that cluster member. The configured weight setting still applies for HTTP requests that do not come through a WebSphere proxy server.

Status

This field indicates whether the cluster member is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

	Started	The server is running.
	Partially stopped	The server is in the process of changing from a started state to a stopped state.
	Stopped	The server is not running.
	Unknown	The server status cannot be determined.

Cluster member settings

Use this page to manage the members of a cluster. A cluster of application servers are managed together and participate in workload management.

A copy of the first cluster member that you create is stored as part of the cluster data and becomes the template for all additional cluster members that you create.

Any individual configuration change that you make to a cluster member does not affect the configuration settings of the cluster member template. You can use `wsadmin` commands to modify the cluster member template, or you can click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members > Templates**. Any change that you make to the template does not affect existing cluster members. See the *Using the administrative clients* PDF for more information on how to use `wsadmin` commands to modify this template.

To view this administrative console page, click **Servers > Clusters > WebSphere application server clusters > *cluster_name***.

On the **Configuration** tab, you can edit fields. You can also click **Installed applications** to view the status of applications that are running on this server. On the **Runtime** tab, which only appears when the cluster member is running, you can look at information about this cluster member. However, the information that displays on this page is read-only. You must return to the **Configuration** tab to change any of the settings that display.

Member name:

Specifies the name of the application server in the cluster. On most platforms, the name of the server is the process name. The member name must match the name of one of the servers that are listed on the application servers page.

Node name:

Specifies the name of the node on which the cluster member is running.

Weight:

Controls the number of requests that are directed to the application server. Even though you specify a value of 0 to 20 as the weight of a server, the weight that is assigned to the server is a proportion, in which, the weight assigned to the server is the numerator, and the sum of the weights of all members of the cluster is the denominator.

When you add a new member to a cluster, the number of requests that are sent to each server in the cluster decreases, assuming that the number of requests coming into the cluster stays the same. Similarly when you remove a new member from a cluster, the number of requests that are sent to each server in the cluster increases, assuming that the number of requests coming into the cluster stays the same.

For example, if you have a cluster that consists of members A, B, and C with weights 2, 3, and 4, respectively, then 2/9 of the requests are assigned to member A, 3/9 are assigned to member B, and 4/9 are assigned to member C. If a new member, member D, is added to the cluster and member D has a weight of 5, then member A now gets 2/14 of the requests, member B gets 3/14 of the requests, member C gets 4/14 of the requests, and member D gets 5/14 of the requests.

On z/OS, weight is used to balance some of the workload types, but others are balanced by the z/OS system.

- For HTTP requests, weights are used to distribute HTTP traffic between the Web server plug-in and the controller handling the clustered application server. Assign a higher weight value to the application server that should receive the HTTP traffic.

- For Web services calls, information is transferred from a servant in one application server to a controller in another application server. The application server that receives the call has the highest weight value.
- Weight has no affect on Internet Inter-ORB Protocol (IIOP) requests. IIOP requests are distributed to the correct application server using the sysplex distributor.

Data type Integer
Range 0 to 20

Unique ID:

Specifies a numerical identifier for the application server that is unique within the cluster. The ID is used for affinity.

Data type Hexadecimal

Short name:

Specifies the short name for this cluster member. This field only displays if you are running on z/OS.

The short name is the default z/OS job name and identifies the cluster member to the native facilities of the operating system, such as Workload Manager (WLM), Automatic Restart Manager, SAF (for example, RACF), started task control, and others.

If you specify a short name for a cluster member, the name:

- Must be one to eight characters in length. By default, when you are running the product on z/OS, the product assumes you are using a 7-character server short name (JOBNAME). If your naming standards require 8 characters, you can lengthen the 7-character server short name to 8 characters.
- Must contain only uppercase alphanumeric characters
- Cannot start with a number.
- Must be unique in the cell.
- Cannot be the same as the value specified on the `ClusterTransitionName` custom property of any non-clustered server. Do not specify a cluster transition name for a server that is part of a cluster.

If you do not specify a short name, the system assigns a default short name that is automatically unique within the cell. You can change the generated short name to conform with your naming conventions.

Data type String

Run in development mode:

Enabling this option may reduce the startup time of an application server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers. This setting is only available on application servers that are running in Version 6.0 and or higher cells.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type Boolean

Default false

Parallel start:

Specifies whether to start the server on multiple threads. When you start the server on multiple threads, the server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the server uses multiple threads when it starts. Setting this option to `false` specifies that the server uses a single thread when it starts, which might lengthen startup time.

The order in which applications start depends on the weight you assign to each application. The application with the lowest starting weight starts first. Applications with the same starting weight start in parallel. Use the `Starting weight` field on the **Applications > Application Types > WebSphere enterprise applications > application_name > Startup behavior** page of the administrative console to set the starting weight for an application.

Data type Boolean
Default true

Start components as needed:

Select this field if you want the cluster member components started as they are needed by an application that is running on this cluster member.

When this property is selected, cluster member components are dynamically started as they are needed. When this property is not selected, all of the cluster member components are started during the cluster startup process. Therefore, selecting this option can improve startup time, and reduce the memory footprint of the cluster members, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the cluster, are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

Note: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

Run in 64 bit JVM mode:

Specifies that the application server runs in 64-bit mode, which is the default setting. Running in 64-bit mode provides additional virtual storage for user applications. This field only displays if you are running on z/OS.

You can deselect this setting if you need to run the application server in 31-bit mode.

There is no interdependence between the modes in which you are running different servers. Therefore, you can run some of your servers in 64-bit mode and some of your servers in 31-bit mode. However, you should eventually convert all of your servers to run in 64-bit mode because support for running servers in 31-bit mode is deprecated.

Access to internal server classes:

Specifies whether the applications that are running on this server can access many of the server implementation classes.

If you select `Allow`, then, applications can access most of the server implementation classes. If you select `Restrict`, then applications cannot access server implementation classes. The applications get a `ClassNotFoundException` error if they attempt to access these classes.

Usually, you should select `Restrict` for this property, because most applications use the supported APIs, and do not need to access any of the internal classes. However, if your application requires the use of one or more of the internal server classes, then select `Allow` as the value for this property.

The default value for this property is `Allow`.

Class loader policy:

Specifies whether there is a single class loader that loads all of the applications, or a different class loader loads each application.

Class loading mode:

Specifies whether the class loader searches in the parent class loader, or in the application class loader first to load a class. The standard for Developer Kit class loaders and product class loaders is `Classes loaded with parent class loader first`.

This field only applies if you set the `Class loader policy` field to `Single`.

If you select `Classes loaded with local class loader first (parent last)`, your application can override classes that are contained in the parent class loader, but this action can potentially result in `ClassCastException`, or linkage errors, if you have mixed use of overridden classes and non-overridden classes.

Process ID:

Specifies the native operating system process ID for this server.

The process ID property is read only. The system automatically generates the value.

Cell name:

Specifies the name of the cell in which this server is running.

The Cell name property is read only.

Node name:

Specifies the name of the node in which this server is running.

The Node name property is read only.

State:

Specifies the runtime state for this server.

The State property is read only.

Cluster member templates collection

Use this page to view the list of cluster member templates that exist for this cluster. To edit the server properties of a template, click the name of that template.

Usually, only one template exists that you can use to create additional members for a cluster. However, if a cluster includes nodes that are at different versions of the product, a different template exists for each of these versions. For example, if a cluster has cluster members residing on both a Version 6.1 node and a Version 7.0 node, the cluster has two templates. The Version 6.1 template is used when you create an additional cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional cluster member on the Version 7.0 node.

If you modify a template, all new cluster members are created with the server property settings of the modified template. However, the property settings of existing cluster members do not change. If you make any change to a cluster member template, you should make the same change to all of the existing cluster members.

To change the server attributes of an existing cluster member, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Clusters members**, and then click the name of the existing cluster member.

To view the cluster member templates that are available for creating a new member of a cluster, in the administrative console, click **Servers > Clusters > WebSphere application server clusters > *cluster_name* > Cluster members > *cluster_member_name***, and then click **Templates**.

Name

Specifies the name of the cluster member template.

Platform

Specifies the operating system platform to which this template applies.

Version

Specifies the version of the product to which the template applies.

Description

Specifies a description of this cluster member template. This field is optional and might be blank.

Starting clusters

You can start all members of a cluster at the same time by requesting that the state of a cluster change to *running*. That is, you can start all application servers in a server cluster at the same time.

Before you begin

Make sure that the members of your cluster have the debug port properly set. If multiple servers on the same node have the same debug port set, the cluster could fail to start. See “Java virtual machine settings” on page 270 for more information on how to change the debug port.

If you want cluster member components to dynamically start as they are needed by the installed applications, verify that the **Start components as needed** option is selected in the configuration settings for each of the cluster members before you start the cluster. Selecting this option can improve cluster startup time, and reduce the memory footprint of the cluster members. Starting components as they are needed is most effective if all of the applications that are deployed on the cluster are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JSPs and Enterprise JavaBeans (EJB).

Note: To ensure compatibility with other WebSphere products, the default setting for this option is deselected. Before selecting this option, verify that any other WebSphere products, that you are running in conjunction with this product, support this functionality.

About this task

When you request that all members of a cluster start, the cluster state changes to partially started and each server that is a member of that cluster launches, if it is not already running. After all members of the cluster are running, the cluster state becomes running.

Note: From the z/OS MVS console, you must individually start each server that you want to run. With the administrative console, you can start each server individually, or you can start a cluster. Starting a cluster automatically starts all of the servers that are defined as part of that cluster.

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters > .**
2. Select the clusters whose members you want started.
3. Click **Start** or **Ripplestart**.
 - **Start** launches the server process of each member of the cluster by calling the node agent for each server to start the servers. After all servers are running, the state of the cluster changes to *running*. If the call to a node agent for a server fails, the server does not start.
 - **Ripplestart** combines stopping and starting operations. It first stops and then restarts each member of the cluster. For example, your cluster contains 3 cluster members named server_1, server_2 and server_3. When you click Ripplestart, server_1 stops and restarts, then server_2 stops and restarts, and finally server_3 stops and restarts. Use the Ripplestart option instead of manually stopping and then starting all of the application servers in the cluster.

Note: If recently added cluster members do not start, you might not have selected **Synchronize changes with nodes** when you added the members to the cluster. To determine if this is the problem:

- a. In the administrative console click **Servers > Clusters > WebSphere application server clusters > ,** select the cluster whose members did not start, and click **Stop**.
- b. Click the name of the cluster, click **OK**, and then click **Review**.
- c. Select **Synchronize changes with Nodes**, and then click **Save**.
- d. Start the cluster and verify that all of the cluster members now start.

Results

When you start the members of a cluster, you automatically enable workload management.

Stopping clusters

Use this task to stop a cluster and any application servers that are members of that cluster.

Before you begin

About this task

You can stop all application servers that are members of the same cluster at the same time by stopping the cluster.

1. Click **Servers > Clusters > WebSphere application server clusters >** in the console navigation tree to access the Server Cluster page.
2. Select those clusters whose members you want stopped.
3. Click **Stop** or **Immediate Stop**.

- **Stop** halts each server in a manner that allows the server to finish existing requests and allows failover to another member of the cluster. When the stop operation begins the cluster state changes to *partially stopped*. After all servers stop, the cluster state becomes Stopped.
- **Immediate Stop** brings down the server quickly without regard to existing requests. The server ignores any current or pending tasks. When the stop operation begins, the cluster state changes to *partially stopped*. After all servers stop, the cluster state becomes Stopped.

Results

All application servers in the sysplex associated with this cluster are issued a request to stop. In addition, a **stop** can be issued against each individual server from the MVS console. To shut down the product environment on a specific system, stop that system daemon. Stopping the system daemon brings down all other server instances on the system. To bring the product down on all of your systems, stop the daemons on all systems. When you stop the location service daemon on one system, it does not bring down the servers on the other systems.

What to do next

See Chapter 5, “Balancing workloads with clusters,” on page 293 for more information about the tasks you can complete with clustering.

Replicating data across application servers in a cluster

Use this task to configure a data replication domain to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans. Data replication domains use the data replication service (DRS), which is an internal component that performs replication services, including replicating data, objects, and events among application servers.

Before you begin

Determine if you are using a multi-broker replication domain. If you configured a data replication domain with a previous version of the product, you might be using a multi-broker replication domain. Any replication domains that you create with the current version of the product are data replication domains. You should migrate any multi-broker replication domains to data replication domains.

About this task

Use this task to configure *replication*, a service that transfers data, objects, or events among the application servers in a cluster. Use replication to prevent loss of session data with session manager, to further improve the performance of the dynamic cache service, and to provide failover in stateful session beans.

Note: If you select the **Configure HTTP memory-to-memory replication** option when you create a cluster, the replication domain is automatically created for you.

Similarly if, instead of WAS01Network , the cell name is simply WAS1, you have to pad the high level qualifier with the first three characters of the string DRSSTREAM. The high level qualifier then becomes WAS1DRS.

Complete the following steps to define two logstreams for an application on which DRS uses RLS for data replication.

1. Create a replication domain. Use one of the following methods to create a replication domain:

- **Create a replication domain manually.**

To create a replication domain manually without creating a new cluster, click **Environment > Replication domains > New** in the administrative console.

On this page you can specify the properties for the replication domain, including timeout, encryption, and number of replicas.

- **Create a replication domain when you create a cluster.**

To create a replication domain when you create a cluster, click **Servers > Clusters > Clusters > New** in the administrative console. Then click **Configure HTTP memory-to-memory replication**. The replication domain that is created has the same name as the cluster and has the default settings for a replication domain. The default settings for a replication domain are to create a single replica of each piece of data and to have encryption disabled. To modify the replication domain properties, click **Environment > Replication domains > New *replication_domain_name*** in the administrative console.

2. Configure the consumers, or the components that use the replication domains. Dynamic cache, session manager, and stateful session beans are the three types of replication domain consumers. Each type of consumer must be configured with a different replication domain. For example, session manager uses one replication domain and dynamic cache uses a different replication domain. However, use one replication domain if you are configuring HTTP session memory-to-memory replication and stateful session bean replication. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

Results

Data is replicating among the application servers in a configured replication domain.

What to do next

If you select DES or 3DES as the encryption type for a replication domain, an encryption key is used for the encryption of messages. At regular intervals, for example once a month, you should go to the **Environment > Replication domains > New** page in the administrative console, and click **Regenerate encryption key** to regenerate the key. After the key is regenerated, you must restart all of the application servers that are configured as part of the replication domain. Periodically regenerating the key improves data security.

Replication

Replication is a service that transfers data, objects, or events among application servers. Data replication service (DRS) is the internal WebSphere Application Server component that replicates data.

Use data replication to make data for session manager, dynamic cache, and stateful session beans available across many application servers in a cluster. The benefits of using replication vary depending on the component that you configure to use replication.

- Session manager uses the data replication service when configured to do memory-to-memory replication. When memory-to-memory replication is configured, session manager maintains data about sessions across multiple application servers, preventing the loss of session data if a single application server fails. For more information about memory-to-memory replication, see the *Administering applications and their environment* PDF.
- Dynamic cache uses the data replication service to further improve performance by copying cache information across application servers in the cluster, preventing the need to repeatedly perform the same tasks and queries in different application servers. For more information about replication in the dynamic cache, see the *Administering applications and their environment* PDF.
- Stateful session beans use the replication service so that applications using stateful session beans are not limited by unexpected server failures. For more information about stateful session bean failover, see the *Developing and deploying applications* PDF.

You can define the number of *replicas* that DRS creates on remote application servers. A replica is a copy of the data that copies from one application server to another. The number of replicas that you configure

affects the performance of your configuration. Smaller numbers of replicas result in better performance because the data does not have to be copied many times. However, if you create more replicas, you have more redundancy in your system. By configuring more replicas, your system becomes more tolerant to possible failures of application servers in the system because the data is backed up in several locations.

Defining a single replica configuration helps you to avoid a single point of failure in the system. However, if your system must be tolerant to more failure, introduce extra redundancy in the system. Increase the number of replicas that you create for any HTTP session that is replicated with DRS. The **Number of replicas** property for any replication domain that is used by the dynamic cache service must be set to Entire domain.

Session manager, dynamic cache, and stateful session beans are the three *consumers* of replication. A consumer is a component that uses the replication service. When you configure replication, the same types of consumers belong to the same *replication domain*. For example, if you are configuring both session manager and dynamic cache to use DRS to replicate objects, create separate replication domains for each consumer. Create one replication domain for all the session managers on all the application servers and one replication domain for the dynamic cache on all the application servers. The only exception to this rule is to create one replication domain if you are configuring replication for HTTP sessions and stateful session beans. Configuring one replication domain in this case ensures that the backup state information is located on the same backup application servers.

See the *Administering applications and their environment* PDF for more information on how to configure replication.

Replication domain collection

Use this page to view the configured replication domains that are used for replication by the HTTP session manager, dynamic cache service, and stateful session bean failover components. All components that need to share information must be in the same replication domain. Data replication domains replace multi-broker replication domains that were available for replication in prior releases. Migrated application servers use multi-broker replication domains which are collections of replicators. You should migrate any multi-broker replication domains to be data replication domains.

To view this administrative console page, click **Environment > Replication domains**.

Name

Specifies a name for the replication domain. The name of the replication domain must be unique within the cell.

Domain type

Following are the two types of replication domains:

Multi-broker domain	Specifies a replication domain that was created with a previous version of WebSphere Application Server. This type of replication domain consists of replicator entries. Support of this type of domain remains for backward compatibility, but is deprecated. Multi-broker and data replication domains do not communicate with each other, so migrate any multi-broker replication domains to the new data replication domains. You cannot create a multi-broker domain or replicator entries in the administrative console after the deployment manager is upgraded to the current version of WebSphere Application Server.
---------------------	--

Data replication domain	Specifies a replication domain created with the latest version of WebSphere Application Server. If the deployment manager has been upgraded to the latest version of WebSphere Application Server, you can create data replication domains only. With the data replication domain, you can specify a number of replicas instead of statically partitioning your replication settings. Specify a data replication domain for each consumer of the domain, for example, two separate domains for dynamic cache and session manager.
-------------------------	---

Data replication domain settings

Use this page to configure a data replication domain. Use data replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster.

To view this administrative console page, click **Environment > Replication domains > replication_domain_name**.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies how long a replication domain consumer waits when requesting information from another replication domain consumer before it gives up and assumes the information does not exist.

Units	seconds
Default	5 seconds

Encryption type:

Specifies the type of encryption to use when transferring replicated data to another area of the network. Select NONE if you don't want to use encryption, DES if you want to use data encryption standard, or 3DES if you want to use triple DES. The default is NONE. The DES and 3DES options encrypt data sent between application server processes (for example, session manager and dynamic caching). Encrypting data improves the security of the network that joins the processes.

If you select DES or 3DES, after you click **Apply** or **OK**, a key for global data replication is generated. At regular intervals, for example once each month, you should navigate to this page in the administrative console and click **Regenerate encryption key** to regenerate this key. Periodically regenerating the key enhances security.

Data type	String
Default	NONE

Number of replicas:

Specifies the number of replicas that are created for every entry or piece of data that is replicated in the replication domain.

Single replica	When you select this option, every HTTP session is replicated to exactly one other application server. This is the default value.
-----------------------	---

Entire domain

When you select this option, each object is replicated to every application server that is configured as a user of the replication domain.

Specify

When you select this option, you must specify, in the Number of replicas field, the number of replicas that you want created for each HTTP session.

Migrating servers from multi-broker replication domains to data replication domains

You can migrate multi-broker replication domains to data replication domains. Any multi-broker domains that exist in your application server environment were created with a previous version of the product.

Before you begin

Determine if the application server configuration you are migrating:

1. Uses an instance of data replication service in peer-to-peer mode or in client/server mode.
Before you begin migrating a client/server mode replication domain, consider if migrating your replication domains might cause a single point of failure. Because you migrate the servers to the new type of replication domain one at a time, you risk a single point of failure if there are 3 or fewer application servers. Before migrating, configure at least 4 servers that use multi-broker replication domains. Perform the following steps to migrate the multi-broker domains to data replication domains:
Dynamic cache replication domains use the peer-to-peer topology.
2. Uses HTTP session memory-to-memory replication that is overloaded at the application or web module level.
If the application server configuration you are migrating uses HTTP session memory-to-memory replication that is overloaded at the application or web module level, you must upgrade your deployment manager to the current version of the product before you start the migration process.

About this task

For HTTP session affinity to continue working correctly when migrating Version 5.x application servers to Version 7.0 application servers, you must upgrade all of the Web server plug-ins for the product to the latest version before upgrading the application servers that perform replication.

After you upgrade your deployment manager to the latest version of the product, you can only create data replication domains. Any multi-broker domains that you created with a previous version of the product are still functional, however, you cannot use the administrative console to create new multi-broker domains or replicators.

The different versions of application servers cannot communicate with each other. When migrating your servers to the current version of the product, keep at least two application servers running on the previous version so that replication remains functional.

Make sure that all of your application servers that are using this multi-broker domain have been migrated to the current version of the product before you start to migrate any multi-broker domains that exist in your configuration.

To migrate the multi-broker domains that exist in your configuration:

1. Migrate two or more of your existing servers to the current version of the product. The remaining servers on the previous version of the product can still communicate with each other, but not with the migrated servers. The migrated servers can also communicate with each other.
2. In the administrative console, create an empty data replication domain. Click **Environment > > Replication domains > New** to create an empty data replication domain.

3. Add two of your migrated servers to the new data replication domain.
For example, if you are migrated four servers, only add two of them to the new replication domain.
4. Configure the two servers as consumers of the replication domain.
Configuring the servers as consumers of the replication domain enables them to use the new domain to share data.
5. Add some of the clients to the new data replication domain.
Perform this step only if the application server configuration you are migrating uses an instance of data replication service in client/server mode.
6. Configure these clients as consumers of the replication domain.
7. Verify that the new data replication domain are successfully sharing data.
Only the servers and clients that are added to the data replication domain and are configured as consumers of this domain can use the data replication domain functions.
8. Add the rest of your migrated servers to the new data replication domain.
When the servers can use the new data replication domain to successfully share data, migrate the rest of the servers that are using the multi-broker replication domain to the new data replication domain.
For example, if you are migrated four servers, add the remaining two servers to the new replication domain.
9. Configure these servers as consumers of the replication domain.
10. Add the rest of the clients to the new data replication domain.
Perform this step only if the application server configuration you are migrating uses an instance of data replication service in client/server mode.
11. Configure these clients as consumers of the replication domain.
12. Restart all of the application servers and clients.
13. Delete the empty multi-broker replication domain.

What to do next

During this process, you might lose existing sessions. However, the application remains active through the entire process, so users do not experience down time during the migration. Create a new replication domain for each type of consumer. For example, create one replication domain for the session manager and another replication domain for dynamic cache.

Data replication domains

Data replication domains and multi-broker domains both perform the same function, which is to provide data replication between application servers in a cluster. Even though you can still configure existing multi-broker domains with the current version of the product, after you upgrade your deployment manager, you can only create data replication domains in the administrative console.

Note: A replication domain that was created with a previous version of the product might be a multi-broker domain. You should migrate these multi-broker domains to data replication domains. Data replication domains enable you to:

- Configure all of the instances of replication that need to communicate in the same replication domain.
- Configure the session manager with both types of replication domains to use topologies such as peer-to-peer, and client-to-server to isolate the function of creating and storing replicas on separate application servers.
- Control the redundancy of replication for each type of replication domain, because with a data replication domain, you can specify a specific number of replicas.

If you used multi-broker domains with earlier releases of the product, use the following comparison chart to learn the differences between how Version 5.x and Version 6.x and Version 7.0 application servers use the two types of replication domains:

	Version 5.x application servers using replication domains	V6.x and Version 7.0 application servers using replication domains
Replication domain types	Uses only multi-broker replication domains for replication.	Servers that are using the current version of the product can be configured to use both multi-broker replication domains and data replication domains for replication. The two types of domains provide backward compatibility with multi-broker domains that were created with a Version 5.x server. You should migrate any multi-broker domains to data replication domains.
Data transport method	Uses multi-broker domain objects that contain configuration information for the internal Java Message Service (JMS) provider, which uses JMS brokers as replicators.	Uses data replication domain objects that contain configuration information to configure the high availability framework on the product. The transport is no longer based on the JMS API. Therefore, no replicators and no JMS brokers exist. You do not have to perform the complex task of configuring local, remote, and alternate replicators. The earlier version of the product did not support data replication domains. The current version of the product can be configured to perform replication using old multi-broker domains by ignoring any JMS-specific configuration and by using the other parameters to configure replication through the high availability framework.
Replication domain configuration	The earlier version of the product encourages the sharing of replication domains between different consumers, such as session manager and dynamic cache.	The current version of the product encourages creating a separate replication domain for each consumer. For example, create one replication domain for session manager and another replication domain for dynamic cache. The only situation where you should configure one replication domain is when configuring session manager replication and stateful session bean failover. Using one replication domain in this case ensures that the backup state information of HTTP sessions and stateful session beans are on the same application servers.

	Version 5.x application servers using replication domains	V6.x and Version 7.0 application servers using replication domains
Partial partitioning	You can configure partial partitioning. Partition the replication domain to filter the number of processes to send data.	Partial partitioning is deprecated. When using data replication domains, you can specify a specific number of replicas for each entry. However, if you specify a number of replicas larger than the number of backup application servers that are running, the number of replicas is the number of application servers that are running. After the number of application servers increases above your configured number of replicas, the number of replicas that are created is equal to the number that you specified.
Domain sharing	Multiple data replication service (DRS) instances share multi-broker domains. A limitation exists on the number of multi-broker domains that you can create because every multi-broker domain contains at least one replicator. A maximum of one replicator can be on each application server.	All DRS instances in a replication domain use the same mode. Each replication domain must contain either client only and server only instances, or client and server instances only. For example, if one instance is configured to client and server, all other instances must be client and server. If one instance in a replication domain is configured to be a client only, you can add client only and server only instances, but not a client and server instance.

Deleting replication domains

You can manually create a replication domain, or have a replication domain automatically generated when you create a cluster. When you no longer need a previously defined replication domain, you can use either the wsadmin AdminConfig delete command or the administrative console to delete the replication domain from your application server environment. Deleting a cluster does not automatically delete a replication domain that is associated with that cluster.

Before you begin

- Verify that none of the applications that you are running require the replication domain you are deleting.
- Verify that neither dynamic caching nor the session manager are configured to use the replication domain that you are deleting. Deleting a replication domain that either dynamic caching or the session manager has been configured to use might cause processing errors.

About this task

Perform the following steps if you want to use the administrative console to delete a replication domain.

1. In the administrative console, click **Environment > > Replication domains**.
2. Select the replication domain that you want to delete.
3. Click **Delete**.
4. Select Synchronize changes with Nodes, and then click **Save** to save your workspace changes to the master configuration.

Results

Data is not replicated amongst the application servers that were part of the configured replication domain that you deleted, and all session data that is associated with the deleted replication domain is deleted.

Replicating data with a multi-broker replication domain

Use this task to manage replication domains that you migrated from a Version 5.x product environment.

Before you begin

Note: Multi-broker replication domains are not created in Version 7.0 product environments. However they can be migrated from existing Version 5.x product environments. If you migrate Version 5.x multi-broker replication domains, you can use the Multi-broker domain panel in the Version 7.0 administrative console to manage these domains.

Although you can manage migrated multi-broker domains with the current version of the product, after you upgrade your deployment manager, you can create only data replication domains in the administrative console. Consider migrating any existing multi-broker domains to the new data replication domains.

About this task

If you are performing this task, it is assumed that you configured replication with a previous version of the product, and defined replication domains that list connected replicator entries, residing in managed servers in the cell,) that can exchange data. You can manage these existing replication domains and replicator entries, but you cannot create new multi-broker replication domains or new replicator entries in the administrative console.

A replicator does not need to run in the same process as the application server that uses it. However, it might be easier to manage replicators and replication domains if a one-to-one relationship exists between replicators and application servers. During configuration, you can select the local replicator as the default replicator.

1. Manage multi-broker replication domain configuration settings. In the administrative console, click **Environment > Replication domains**.
2. Click **Multi-broker domain > *multi-broker domain_name***, and update the values for that particular multi-broker replication domain. The default values are generally sufficient, especially for the pooling and timeout properties.
 - a. Name the replication domain.
 - b. Specify the timeout interval.
 - c. Specify the encryption type. The DES and TRIPLE_DES options encrypt data sent between application server processes and better secure the network joining the processes.
 - d. Partition the replication domain to filter the number of processes to which data is sent. Partitioning the replication domain is most often done if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data that is maintained by Web container dynamic caching.
 - e. Specify whether you want a single replication of data to be made. Enable the option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails.
 - f. Specify whether processes should receive data in objects or bytes. Processes receiving data in objects receive the data and class definitions. Processes receiving data in bytes receive the data only.
 - g. Configure a pool of replication resources. Pooling replication resources can enhance the performance of the replication service.

3. Maintain the replicators that you have already defined. You cannot create any new replicators. The default convention is to define a replicator in each application server that uses replication. However, you can define a pool of replicators, separate from the servers hosting applications.
 - a. In the administrative console, click **Environment > Replication domains** *replication_domain_name* > **Replicator entries** > *replicator_entry_name*.
 - b. Specify a replicator name and select a server available within the cell to which you can assign a replicator. Also specify a host name and ports. Note that a replicator has two ports (replicator and client ports) that use the same host name but have different ports.
4. Click **OK** to save your changes.

What to do next

If you use the DES or TRIPLE_DES encryption type for a replicator, at regular intervals, such as monthly, you should click **Regenerate encryption key** on the Replication domains settings page. Periodically changing the encryption key enhances security.

Multi-broker replication domains

A multi-broker replication domain is a collection of replication entries, or *replicator* instances, used by clusters or individual servers within a cell. Multi-broker replication domains were created with a previous release of the product.

Note: After you upgrade your deployment manager to the latest version of the product, you can create data replication domains only. Any multi-broker domains that you created with a previous release of the product are still functional, however, you cannot create new multi-broker domains or replicator instances with the administrative console.

A replication entry, or replicator, is a run-time component that handles the transfer of internal product data. All replicators within a replication domain connect with each other, forming a network of replicators.

Components such as session manager and dynamic cache can connect to any replicator within a domain to receive data from their peer components on other application servers that are connected other replicators in the same domain. If the replicator that a component is connected to fails, the component automatically attempts to reconnect to another replicator in the domain and recover any data that was missed while the component was not connected to a replicator.

The default is to define a replication domain for a cluster when creating the cluster. However, replication domains can span across clusters.

Global default settings apply to a given replication domain across a cell. Most default settings tune and control the behavior of replicator entries that are in managed servers across the cell. Such default settings control the use of encryption or the serialization and transferring of objects. Some default settings tune and control how specific product functions, such as session manager and dynamic caching, leverage replication, such as session use of partitions.

For situations that require settings values other than the default, change the values for a given replication domain. Settings include various resource allocation, replication strategies, such as grouping or partitioning, and methods, as well as some security related items.

If you are using replication for HTTP session failover, you might also need to filter where the session replicates. For example, only replicate to two places out of many. The global default settings define the partition size or number of groups and the session manager settings define the groups to which a particular instance belongs.

Filtering is less important if you are using replication to distribute information on data that is no longer valid and actual cached data maintained by dynamic caching. Replication does not occur for failover as much

as for data synchronization across a cluster or cell when you likely want to avoid expensive costs for generating data potentially needed across those various servers.

Note that you can filter or segment by using multiple replication domains.

Multi-broker replication domain settings

Use this page to configure a multi-broker replication domain. This administrative console page applies only to replication domains that were created with a previous version of the product. Replication domains use the data replication service (DRS).

To view this administrative console page, click **Environment > Replication domains > *multibroker_replication_domain_name***.

An application server that is connected to a replicator within a domain can access the same set of data sent out by any application server connected to any other replicator, including the same replicator. Data is not shared across replication domains.

Name:

Specifies a name for the replication domain. The name must be unique within the cell.

Request timeout:

Specifies the number of seconds that a replication domain consumer waits when requesting information from another replication domain consumer before giving up and assuming the information does not exist. The default is 5 seconds.

Data type	Integer
Units	Seconds
Default	5

Encryption type:

Specifies the type of encryption used before the object transfers over the network. The options include NONE, DES, TRIPLE_DES. The default is NONE. The DES and TRIPLE_DES options encrypt data sent between application server processes and secure the network joining the processes.

If you specify DES or TRIPLE_DES, a key for global data replication is generated after you click **Apply** or **OK**. When you use the DES or TRIPLE_DES encryption type, click **Regenerate encryption key** at regular intervals such as monthly because periodically changing the key enhances security.

DRS partition size:

Specifies the number of groups into which a replication domain is partitioned. By default, data sent by an application server process to a replication domain is transferred to all other application server processes connected to that replication domain. To filter or reduce the number of destinations for the data being sent, partition the replication domain. There should be at least one server listening to every partition. If there are no servers listening on a partition, all the replicas created in that partition are lost because there is no server to cache the objects. The default partition size is 10, and the partition size should be 10 or more to enhance performance.

Partitioning the replication domain is only applicable if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. Partitioning is not supported for sharing of cached data maintained by Web container dynamic caching. As to dynamic caching, all partitions or groups are always active and used for data replication.

When you partition a replication domain, you define the total number of groups or partitions. Use this setting to define the number of groups. Then, when you configure a specific session manager under a Web container or as part of an enterprise application or Web module, select the partition to which that session manager instance listens and from which it accepts data. To specify the groups to which an application server listens, change the settings for affected servers on a session manager page. In addition, you can set a role or runtime mode for a server. This role or mode affects whether an application server process sends data to the replication domain, receives data, or does both. The default is both to receive and send data.

Data type	Integer
Default	10

Single replica:

Specifies that a single replication of data is made. Use this option only if you are using session manager with memory to memory replication. Enable this option if you are replicating data to support retrieval of an HTTP session if the process maintaining the HTTP session fails. This option restricts the recipient of the data to a single instance.

Note: Do not enable this option on a domain that is using dynamic cache replication. This setting provides filtering beyond grouping or partitioning. Using this setting, you can choose to have data only sent to one other listening instance in the replication domain.

Default	false
----------------	-------

Serialization method:

Specifies the object serialization method to use when replicating data. An administrative concern with replicating Java objects is locating the class definition, especially in a Java Platform, Enterprise Edition (Java EE) environment where class definitions might reside only in certain web modules or enterprise applications. Object serialization methods define whether the processes receiving data also need the class definition.

The options for this setting are OBJECT and BYTES. The default is BYTES.

OBJECT instructs a replicator to write the object directly to the stream. With OBJECT, a replicator must instantiate the object on the receiving side so it must have the class definition.

BYTES instructs a replicator to break down the object into bytes and then send only the bytes across the stream. With BYTES, a replicator does not need to instantiate the object on the receiving side. The BYTES option is useful for failover, where the data is not used at the receiving side and the class definitions do not need to be stored on the receiving side. Or, the option requires that you move class definitions from the Web application class path to the system class path.

DRS pool size:

Specifies the size of the pool of resources allocated for communication with its Java Message Service (JMS) transport. You must configure this number to be the same as the DRS partition size. The default is 10.

Pooling replication resources can enhance the performance of the internal data replication service.

DRS pool connections:

Specifies that the domain replication service should create a pool of connections with its Java Message Service (JMS) transport rather than reusing a single connection. You can pool connections when using a single replica or client server environment. You should not pool connections in a peer to peer environment.

The default is to not create a pool of connections for replication.

Replicator entry collection:

Use this page to view and manage replicator entries. Replicator entries are for use only with multi-broker replication domains. Each multi-broker replication domain consists of one or more replicator entries.

To view this administrative console page, click **Environment > Replication domains > replication_domain_name > Replicator entries**.

Replicator entries are only valid for multi-broker domains, which are replication domains created with a previous version of the product. When you migrate your deployment manager to the current version of the product, you are no longer be able to create new replicator entries in the administrative console. You can only view and modify settings for replicator entries that were created with the previous version of the product.

Replicator name:

Specifies a name for the replicator entry.

Replicator entry settings:

Use this page to view and configure a replicator entry, or *replicator*. Replicators are used with multi-broker replication domains.

To view this administrative console page, click **Environment > Replication domains > replication_domain_name > Replicator entries > replicator_entry_name**.

Replicators communicate using Transmission Control Protocol/Internet Protocol (TCP/IP). Therefore, you must allocate an IP address and ports for replicators. Use this page to name a replicator and then to allocate an IP address and two ports (replicator and client ports) for the replicator.

Replicator name:

Specifies a name for the replicator entry.

Server:

Specifies the server for which you are defining a replicator. You can view the names of servers that do not already have replicators. You can create a maximum of one replicator on any application server.

Replicator and client host name:

Specifies the IP address, domain name service (DNS) host name with domain name suffix, or just the DNS host name, used by a client to request a Web application resource (such as a servlet, JavaServer Pages (JSP) file, or HTML page).

A replicator port and client port share the same host name.

Replicator Port:

Specifies the port for which the replicator is configured to accept messages from other replicators. The port value is used in conjunction with the host name.

The replicator port enables communication among replicators. It provides replicator port to replicator communication. The usual value specified is 7874.

Client Port:

Specifies the port for which the Web server is configured to accept client requests. The port value is used in conjunction with the host name.

The client port enables communication between an application server process and a replicator. It provides client port to application server communication. The usual value specified is 7873.

Deleting clusters

Use this task to remove a cluster and all of its cluster members.

Before you begin

Removing a cluster deletes the cluster and all associated cluster members. When you delete a cluster, there is no option to keep certain cluster members or applications that you have installed on any part of the cluster.

In a production environment, avoid deleting clusters that are carrying workload. You can, however, add and remove cluster members during production. When you want to remove a cluster, create a new cluster, adding new members while the old cluster is still operational. After the new cluster is working, remove the cluster members from the old cluster and then delete the cluster.

Note: If the cluster you are removing has applications or modules mapped to it, remap the modules to another cluster, or create a new cluster and remap the modules to the new cluster, before removing the old cluster. After a cluster to which modules are mapped is deleted, the modules cannot be remapped to another cluster. Therefore, if you do not remap the modules to another cluster before deleting the old one, you must uninstall all of the modules that were mapped to the old cluster, and then reinstall them on a different cluster.

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters**.
2. Make sure the cluster you want to remove is stopped.
If the cluster is started, stop the cluster.
3. Delete the cluster. Select the cluster you want to delete, and click **Delete**.
4. Click **OK** and then click **Review** to preview your changes.
5. Select **Synchronize changes with Nodes**, and then click **Save** to save your changes.

Results

The cluster and all of the cluster members are deleted.

Deleting specific cluster members

Use this task to remove a cluster member from an existing cluster. Removing a cluster member deletes the associated application server.

About this task

You must delete an application server to remove it from a cluster.

If, in the administrative console, you select **Include cluster members in the collection** as one of your console page preferences for the Application servers page, you can use either the Application servers page or the Cluster members page to delete an application server.

To use the Cluster members page to remove an application server from a cluster:

1. In the administrative console, click **Servers > Clusters > WebSphere application server clusters > .**
2. Click the name of the cluster that contains the cluster member that you are removing from the cluster, and then click **Cluster members**.
3. Check the status of the cluster member that you are removing. If the cluster member is started, select the cluster member, and click **Stop**, and then view the Status of this cluster member again, along with any messages or logs to make sure the cluster member stops. You cannot remove a cluster member while it is running.
4. Delete the cluster member. Select the cluster member you want to delete, and click **Delete**.
5. Click **OK** and then click **Review** to preview your changes.
6. Select **Synchronize changes with Nodes**, and then click **Save** to save your changes.

Results

The cluster member is deleted.

Configuring an application server to use the WLM even distribution of HTTP requests function

By configuring your application server to use the WLM even distribution of HTTP requests function, HTTP session objects can be evenly distributed by workload management (WLM) to the servants in your configuration. You can use this task to distribute HTTP session objects in a round-robin manner among several servants instead of the normal situation where there is a servant affinity, and HTTP session objects reside in one or two servants.

Before you begin

Your application server should be running on a z/OS system that is at Version 1.4 or later. Because you are distributing HTTP requests among multiple servants in this task, you should also have multiple servants enabled to use this function. See “Enabling multiple servants on z/OS” on page 302 for more information.

About this task

Use this task if your application server is experiencing problems with the default workload distribution strategy. The default workload distribution strategy uses a hot servant for running requests that create HTTP session objects. Consider configuring the product and the z/OS Workload Manager to distribute your HTTP session objects in a round-robin manner in the following conditions:

- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP sessions in memory last for many hours or days.
- A large number of clients with HTTP session objects must be kept in memory.
- The loss of a session object is disruptive to the client or server.
- There is a large amount of time between requests that create HTTP sessions.

For more background about when to use this task, see “WLM even distribution of HTTP requests” on page 365.

1. In the administrative console, set the **WLMStatefulSession** property to true.

- a. Click **Servers > Server Types > WebSphere application servers**, and then select the server for which you want to use the WLM even distribution of HTTP requests function.
 - b. Under Server Infrastructure, click **Servers > Server Types > WebSphere application servers**, and then, under Additional Properties, click **Custom properties**.
 - c. Click **WLMStatefulSession** and change the value in the Value field to true if it is currently set to false. The default value for this property is false for an application server and true for a deployment manager.
 - d. Click **Apply** and then click **Save and synchronize** to update your changes.
2. Set the optimal minimum and maximum number of servants for the workload. Set the minimum and maximum number of servants to handle the expected number of HTTP sessions with affinity. The minimum number of servants should be greater than one. If, for example, you expect 15,000 HTTP session objects are established in the server during the day, then you might set the minimum number of servants to some value larger than one. The minimum of servants is dependent upon the size and number of the HTTP session objects. However, the initial arrival rate of client requests establishing the affinity, the frequency of client interaction, the duration of each client interaction (CPU time and thread occupancy time), and the length of time that the HTTP session object is maintained also need to be considered when establishing the minimum value for the number of servants.
 - a. To set the number of servants, click **Servers > Server Types > WebSphere application servers *server_name* Server instance**.
 - b. Set the minimum and maximum number of servants.
 - c. Click **Save and synchronize** to apply the changes.
 3. If you use a classification mapping file instead of a common workload classification document, and you specify more than one transaction class on a mapping rule for the managed round robin support that the product provides, you should remove this section from your classification mapping file. You should use a common workload classification document instead of a classification mapping file because support for the classification mapping file is deprecated. However if you use a classification mapping file, and that file contains a line similar to the following:

```
TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1 TCLASS2 TCLASS3
```

Modify this line such that it specifies only one transaction class. For example, you might change the preceding line to the following line:

```
TransClassMap *:8080 /Dynacache1Web1/Servlet1 TCLASS1
```

You also must update the z/OS workload manager policy to remove the extra service classes that are only required if you want to use the managed round robin support that the product provides. Following is an example of how to remove the extra service classes:

```
Subsystem-Type Xref Notes Options Help
-----
                Modify Rules for the Subsystem Type          Row 9 to 16 of 16
Command ==> _____ SCROLL ==> CSR

Subsystem Type . : CB          Fold qualifier names?   Y (Y or N)
Description . . . Component Broker requests

Action codes:  A=After      C=Copy      M=Move      I=Insert rule
                B=Before    D=Delete row R=Repeat  IS=Insert Sub-rule
                                   More ==>

Action  -----Qualifier-----          -----Class-----
Type   Name      Start          Service      Report
_____  _____  _____  _____  _____
_____ 1  CN          AZSR01      _____  _____
_____ 2  TC          TCLASS1     _____  _____
_d_____ 2  TC          TCLASS2     _____  _____
_d_____ 2  TC          TCLASS3     _____  _____
_____ 1  CN          AZSR02      _____  _____
_____ 1  CN          AZSR02      _____  _____
***** BOTTOM OF DATA *****
```


- Restart the server. The server recognizes the WLMStatefulSession property after it is restarted.

Results

The application server uses the WLM even distribution of HTTP requests function to handle its workload instead of showing affinity to a certain servant.

What to do next

See “Detecting and handling problems with runtime components” on page 196 to handle problems with server clusters and workloads.

WLM even distribution of HTTP requests

The z/OS workload management (WLM) component supports distributing incoming HTTP requests without servant affinity in a round robin manner across the servants. This functionality is intended for, but not limited to long lasting HTTP session objects that are maintained in memory, stateless session Enterprise JavaBeans (EJB), and the create method for stateful session enterprise beans. You can configure the product to use this functionality to spread HTTP requests among active servants that are currently bound to the same work queue as the inbound requests.

Background

The following diagram represents one clustered server instance. The azsr01 cluster contains the azsr01a application server instance. In the application server instance is a controller, the workload manager (WLM) queue, and the servants where applications run. The controller is the HTTP and IIOOP termination point. The WLM queue controls the flow of work from the controller to one of the servants. Each of the servants contains worker threads that select work from the WLM queue.

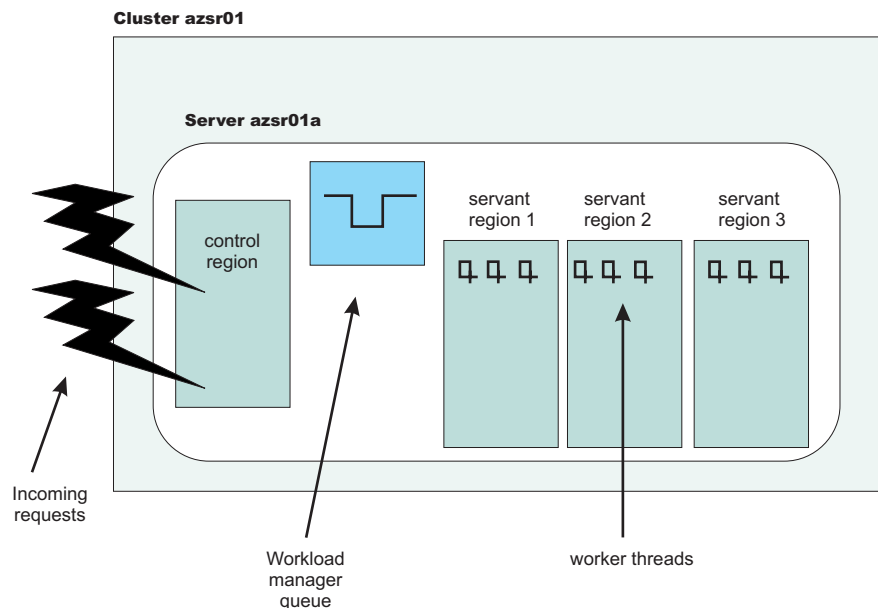


Figure 2. The contents of one clustered server instance

In the preceding diagram, the application server is configured to have the minimum and maximum number of servants set to three.

There are WLM definitions for the application servers in this cluster. All of the requests for any application server instance in the azsr01 cluster are assigned to the same service class. The WLM classification rules assign all enclaves that are running in the azsr01a application server to the AZAMS1 service class. See the following diagrams for an example of the WLM service class definition and the classification rules.

```

Service-Class Xref Notes Options Help
-----
Command ==> Modify a Service Class Row 1 to 2 of 2

Service Class Name . . . . . : AZAMS1
Description . . . . . : WAS Enclave Work
Workload Name . . . . . : ONL_WKL (name or ?)
Base Resource Group . . . . . : _____ (name or ?)
Cpu Critical . . . . . : NO (YES or NO)

Specify BASE GOAL information. Action Codes: I=Insert new period,
E=Edit period, D=Delete period.

---Period--- -----Goal-----
Action # Duration Imp. Description
-----
 1 1 Execution velocity of 50
***** Bottom of data *****

```

Figure 3. The WLM service class definition

```

Subsystem-Type Xref Notes Options Help
-----
Command ==> Modify Rules for the Subsystem Type Row 11 to 20 of 20
SCROLL ==> CSR

Subsystem Type . . : CB Fold qualifier names? Y (Y or N)
Description . . . : Component Broker requests

Action codes: A=After C=Copy M=Move I=Insert rule
              B=Before D=Delete row R=Repeat IS=Insert Sub-rule
              More ==>

Action -----Qualifier----- -----Class-----
Action Type Name Start Service Report
-----
 1 CN AZSR01 _____ DEFAULTS: AZAMS1 RBBDEFLT
 1 CN AZSR02 _____ AZAMS1 RAZAMS1
 1 CN AZSR03 _____ AZAMS2 RAZAMS2
 1 CN AZSR03 _____ AZAMS3 RAZAMS3
***** BOTTOM OF DATA *****

```

Figure 4. The WLM CB subsystem classification rules

The hot servant strategy

The product supports the use of HTTP session objects in memory for application servers with multiple servants. In the following diagram, two users accessed an application in the azsr01a application server instance. User 1 established an HTTP session object in servant 3. User 2 established an HTTP session object in servant 2.

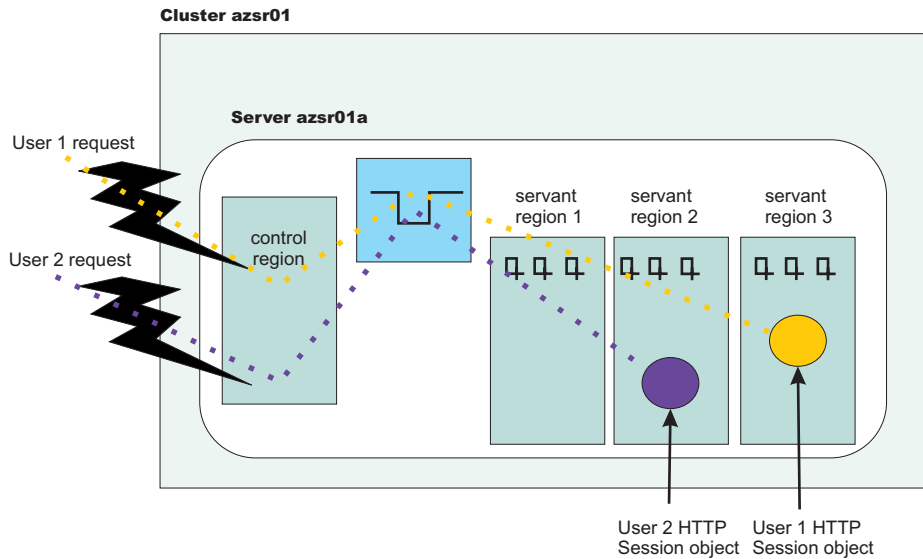


Figure 5. Users establish HTTP session objects

When a user accesses a servant region without an established HTTP session object, no servant region affinity exists. Therefore, the request can be dispatched to any servant that is available. WLM might start a new servant if all of the following conditions exist:

- The configuration allows creating new servants
- The workload manager logic determines that the system can sustain an additional servant
- Adding another servant leads to reduced queue delay and allows enclaves to be completed within the specified goal

When multiple servants are bound to the same service class, WLM attempts to dispatch the new requests to a *hot* servant. A *hot* servant has a recent request dispatched to it and has threads available. If the *hot* servant has a backlog of work, WLM dispatches the work to another servant.

Normally running this *hot* servant strategy is good because the *hot* servant likely has all its necessary pages in storage, has the just-in-time (JIT) compiled application methods saved close by, and has a cache full of data for fast data retrieval. However, this strategy presents a problem in the following situations:

- HTTP session objects in memory are used, causing dispatching affinities.
- The HTTP session objects last for many hours or days.
- A large number of clients with HTTP session objects that must be kept in memory.
- The loss of a session object is disruptive to the client or server and the amount of time between requests that create HTTP sessions is large.

In the last situation, an undesired skew in the distribution of HTTP session objects is the result. In the following diagram, most of the HTTP session objects were assigned to servant 1.

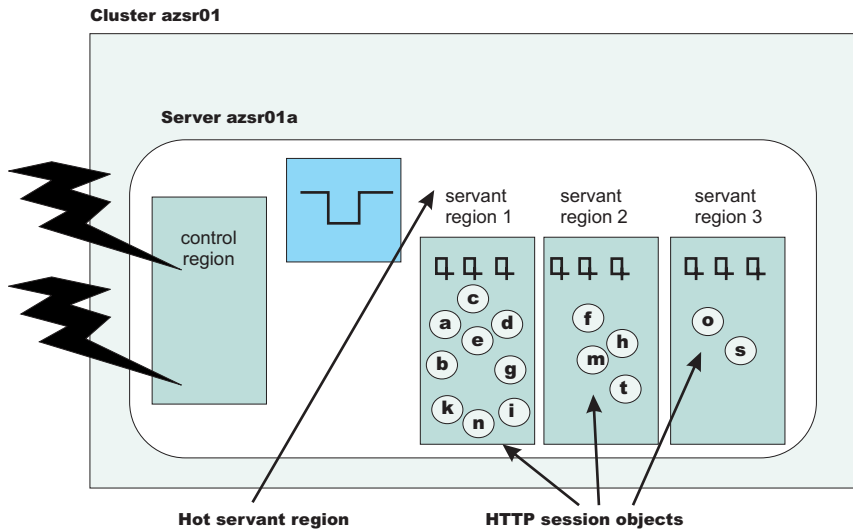


Figure 6. HTTP Session objects assigned to a hot servant

A large percentage of HTTP session objects reside in one or two servants because most of the time, there are not enough requests in the WLM queue to warrant dispatching work among many servants. This behavior can lead to the following undesirable results.

- If the application creates a large number of objects in a single servant, long garbage collection times might result.
- If all the HTTP session objects are bound to one servant, requests might be held in the queue for a long time because the work cannot be managed by WLM and cannot be dispatched in any servant.
- If all HTTP session objects reside in one or two servants, a timeout in a single servant can affect a larger number of users than if the HTTP session objects are divided equally among several servants.

Distribute incoming HTTP requests without servant affinity

If your configuration experiences one of the described situations that cause a problem with the *hot* servant strategy, you can configure your application server to support the distribution of incoming HTTP requests across servants without servant affinity. When you enable this functionality, the application server uses a round-robin distribution of HTTP requests to the servants.

In the following example, assume that the application server was configured to use the round-robin distribution of HTTP requests among the servants and multiple servants are started for the work queue requests that have the same service class assigned.

When a new HTTP request without affinity arrives on a work queue, the WLM checks to see if there is a servant that has at least one worker thread waiting for work. If there are no available worker threads in any servants, WLM queues the request until a worker thread in any of the servants becomes available. If there are available worker threads, WLM finds the servant with the smallest number of affinities. If there are servant regions with equal number of affinities, then WLM dispatches the work to the servant region with the smaller number of busy server threads.

The goal of this algorithm is for WLM to balance the incoming requests without servant affinity among waiting servants while considering changing conditions. The algorithm does not blindly assign requests to servers in a true round-robin manner. The following diagram shows the balanced distribution of HTTP session objects across servants.

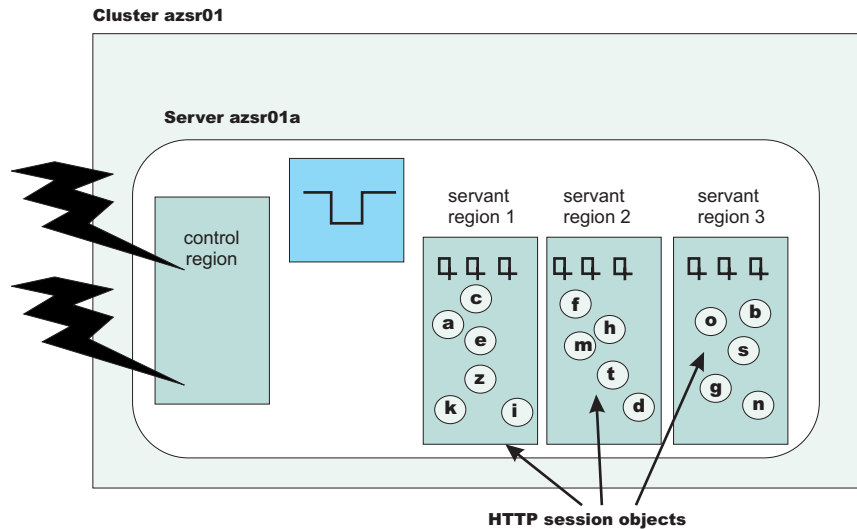


Figure 7. HTTP Session objects assigned to servants without affinity

This distribution mechanism works for all inbound requests without affinity. After the HTTP session object is created, all the client requests are directed to that servant until the HTTP session object is removed.

If you decide to enable the distribution of incoming HTTP requests without servant affinity, you might need to make some changes to your classification mapping file. If you have set up your classification mapping file to specify more than one transaction class on a mapping rule for the managed round robin support that the product provides, you should remove this section from your classification mapping file.

WLM dynamic application environment operator commands

The dynamic application environments are displayed and controlled separately from static application environments. In order to control the dynamic environments, you must set the Resource Access Facility (RACF) server class profiles to give you the proper permission to issue MVS console commands.

If you have the proper permission, you can issue the following commands from the MVS console:

Display a specific dynamic application environment

```
D WLM,DYNAPPL=appl_env_name (appl_env_name is the short cluster name)
```

Display all dynamic application environments

```
D WLM,DYNAPPL=*
```

Restart a specific dynamic application environment

```
V WLM,DYNAPPL=appl_env_name,RESUME
```

Quiesce a specific dynamic application environment

```
V WLM,DYNAPPL=appl_env_name,QUIESCE
```

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories infer specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations (z/OS)

app_server_root

Refers to the top directory for a WebSphere Application Server node.

The node may be of any type—application server, deployment manager, or unmanaged for example. Each node has its own *app_server_root*. Corresponding product variables are `was.install.root` and `WAS_HOME`.

The default varies based on node type. Common defaults are *configuration_root*/AppServer and *configuration_root*/DeploymentManager.

configuration_root

Refers to the mount point for the configuration file system (formerly, the configuration HFS) in WebSphere Application Server for z/OS.

The *configuration_root* contains the various *app_server_root* directories and certain symbolic links associated with them. Each different node type under the *configuration_root* requires its own cataloged procedures under z/OS.

The default is `/wasv7config/cell_name/node_name`.

plug-ins_root

Refers to the installation root directory for Web Server plug-ins.

profile_root

Refers to the home directory for a particular instantiated WebSphere Application Server profile.

Corresponding product variables are `server.root` and `user.install.root`.

In general, this is the same as *app_server_root*/profiles/*profile_name*. On z/OS, this will be always be *app_server_root*/profiles/default because only the profile name "default" is used in WebSphere Application Server for z/OS.

smpe_root

Refers to the root directory for product code installed with SMP/E.

The corresponding product variable is `smpe.install.root`.

The default is `/usr/lpp/zWebSphere/V7R0`.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Intellectual Property & Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA
Attention: Information Requests

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

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 is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.