



## Setting up the application serving environment

**Note**

Before using this information, be sure to read the general information under “Notices” on page 345.

**Compilation date: December 2, 2004**

**© Copyright International Business Machines Corporation 2004. All rights reserved.**

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

---

# Contents

<b>How to send your comments.</b>	ix
<b>Chapter 1. Overview for setting up application serving environments</b>	1
Setting up WebSphere Application Server products	1
Introduction: System administration	2
Introduction: Administrative console	2
Introduction: Administrative scripting (wsadmin)	3
Introduction: Administrative commands	4
Introduction: Administrative programs.	4
Introduction: Administrative configuration data	4
Welcome to basic administrative architecture	4
Introduction: Servers	5
Introduction: Application servers	6
Introduction: Web servers	7
Introduction: Clusters	8
Introduction: Environment	9
Introduction: Cell-wide settings	9
<b>Chapter 2. How do I administer applications and their environments?</b>	11
<b>Chapter 3. Setting up the application serving environment</b>	13
<b>Chapter 4. Planning the installation (diagrams)</b>	15
Planning to install Network Deployment	16
Planning to install Web server plug-ins	25
Planning to install WebSphere Application Server Clients	31
Planning to create application server environments	32
Example: Choosing a topology for better performance	33
Queuing network.	34
Queuing and clustering	35
Queue configuration tips	35
<b>Chapter 5. Configuring the product after installation</b>	39
firststeps command	40
Using the Profile creation wizard	45
Using the Profile creation wizard to create a deployment manager	49
Using the Profile creation wizard to create a custom profile	60
Using the Profile creation wizard to create an application server	73
Deleting a profile.	85
wasprofile command	86
Introduction to terms that describe Version 6 profiles	86
Location of the command file	89
Logging	89
Required disk space	89
Concurrent profile creation	90
Entering lengthy commands on more than one line	90
wasprofile.sh command syntax	90
wasprofile.bat command syntax	91
Parameters.	92
Use case scenarios.	94
Using the installation verification test	100
ivt command	101

<b>Chapter 6. Configuring ports</b> . . . . .	103
Port number settings in WebSphere Application Server versions . . . . .	103
<b>Chapter 7. Communicating with Web servers.</b> . . . . .	107
Web server plug-in properties settings . . . . .	109
Plug-in log file name . . . . .	109
Plug-in installation location . . . . .	109
Plug-in configuration file name . . . . .	110
Automatically generate plug-in configuration file . . . . .	110
Automatically propagate plug-in configuration file . . . . .	111
Ignore DNS failures during Web server startup . . . . .	111
Refresh configuration interval . . . . .	111
Plug-in logging . . . . .	111
Web server plug-in request and response optimization properties settings . . . . .	112
Web server plug-in caching properties settings . . . . .	114
Web server plug-in request routing properties settings . . . . .	115
Web server plug-in custom properties . . . . .	116
Web server plug-in configuration service properties settings . . . . .	117
Enable automated Web server configuration processing . . . . .	117
Application Server property settings for a Web server plug-in . . . . .	117
Server role . . . . .	117
Connect timeout . . . . .	118
Maximum number of connections that can be handled by the Application Server . . . . .	118
Use extended handshake to check whether Application Server is running . . . . .	118
Send the header "100 Continue" before sending the request content . . . . .	119
Web server plug-in configuration properties . . . . .	119
Web server plug-in connections . . . . .	121
Web server plug-in remote user information processing . . . . .	122
Web server plug-ins . . . . .	122
Checking your IBM HTTP Server version . . . . .	123
Web server tuning parameters . . . . .	123
Gskit install images files . . . . .	127
Plug-ins: Resources for learning . . . . .	127
Web server plug-in tuning tips . . . . .	128
Tuning Web servers . . . . .	129
<b>Chapter 8. Setting up the administrative architecture</b> . . . . .	131
Cells . . . . .	131
Configuring cells . . . . .	131
IP version considerations for cells . . . . .	132
Cell settings . . . . .	134
Deployment managers . . . . .	135
Configuring deployment managers . . . . .	135
Running the deployment manager with a non-root user ID . . . . .	135
Deployment manager settings . . . . .	137
Node . . . . .	138
Managing nodes . . . . .	139
Node collection . . . . .	141
Add managed nodes . . . . .	143
Node installation properties . . . . .	144
Node group . . . . .	145
Node group membership rules . . . . .	145
Examples: Using node groups . . . . .	146
Managing node groups . . . . .	147
Node group collection . . . . .	147
Managing node group members . . . . .	148

Node group member collection . . . . .	149
Administration service settings . . . . .	150
Standalone . . . . .	150
Preferred Connector . . . . .	150
Extension MBean Providers collection . . . . .	150
Extension MBean collection . . . . .	151
Java Management Extensions connector properties . . . . .	151
Java Management Extensions connectors . . . . .	156
Repository service settings . . . . .	157
Node agents . . . . .	157
Managing node agents . . . . .	157
Node agent collection . . . . .	157
Remote file services . . . . .	159
Configuring remote file services . . . . .	160
File transfer service settings . . . . .	160
File synchronization service settings . . . . .	161
Administrative agents: Resources for learning . . . . .	163
<b>Chapter 9. Configuring the environment.</b> . . . . .	165
Virtual hosts . . . . .	165
Why you would use virtual hosting . . . . .	165
The default virtual host (default_host) . . . . .	166
How requests map to virtual host aliases . . . . .	166
Configuring virtual hosts . . . . .	168
Virtual host collection . . . . .	168
Variables . . . . .	171
Configuring WebSphere variables . . . . .	172
WebSphere variables collection . . . . .	173
IBM Toolbox for Java JDBC driver . . . . .	174
Configure and use the jt400.jar file. . . . .	175
Shared library files . . . . .	175
Managing shared libraries . . . . .	175
Creating shared libraries . . . . .	176
Shared library collection . . . . .	176
Associating shared libraries with applications . . . . .	177
Associating shared libraries with servers . . . . .	178
Installed optional packages . . . . .	178
Using installed optional packages . . . . .	180
Library reference collection . . . . .	181
Environment: Resources for learning . . . . .	182
<b>Chapter 10. Working with server configuration files</b> . . . . .	183
Configuration documents . . . . .	183
Configuration document descriptions . . . . .	184
Object names . . . . .	186
Configuration repositories . . . . .	187
Handling temporary configuration files resulting from session timeout . . . . .	187
Changing the location of temporary configuration files . . . . .	188
Changing the location of backed-up configuration files . . . . .	188
Changing the location of temporary workspace files . . . . .	188
Backing up and restoring administrative configurations . . . . .	189
Transformation of configuration files . . . . .	189
Server configuration files: Resources for learning . . . . .	189
<b>Chapter 11. Administering application servers</b> . . . . .	191
Application servers . . . . .	191

Understanding server templates. . . . .	191
Creating application servers . . . . .	192
Configuring application servers for UTF-8 encoding . . . . .	193
Creating server templates . . . . .	193
Listing server templates. . . . .	193
Deleting server templates . . . . .	194
Managing application servers . . . . .	194
Server collection . . . . .	195
Starting servers. . . . .	203
Running application servers from a non-root user . . . . .	204
Running an application server from a non-root user and the node agent from root . . . . .	206
Running an Application Server and node agent from a non-root user . . . . .	207
Detecting and handling problems with run-time components . . . . .	209
Stopping servers . . . . .	209
Core group service settings . . . . .	210
Creating generic servers . . . . .	211
Starting and terminating generic servers . . . . .	212
Configuring transport chains . . . . .	213
Transport chains . . . . .	213
HTTP transport channel custom property . . . . .	214
HTTP Tunnel transport channel custom property . . . . .	215
Troubleshooting transport chain problems . . . . .	216
Configuring HTTP transports . . . . .	216
Transport chains collection . . . . .	222
Transport chain settings . . . . .	222
HTTP tunnel transport channel settings . . . . .	223
HTTP transport channel settings . . . . .	223
TCP transport channel settings . . . . .	225
DCS transport channel settings . . . . .	227
Web container transport channel settings . . . . .	228
Custom services . . . . .	228
Developing custom services . . . . .	229
Custom service collection . . . . .	230
Process definition . . . . .	232
Defining application server processes . . . . .	232
Process definition settings. . . . .	232
Automatically restarting server processes . . . . .	235
Java virtual machines (JVMs) . . . . .	243
Using the JVM . . . . .	244
Java virtual machine settings. . . . .	244
Configuring JVM sendRedirect calls to use context root . . . . .	248
Setting custom JVM properties . . . . .	248
Tuning Java virtual machines. . . . .	250
Preparing to host applications . . . . .	251
Java memory tuning tips . . . . .	251
Configuring multiple network interface card support . . . . .	255
Tuning application servers. . . . .	256
Web services client to Web container optimized communication . . . . .	258
Application servers: Resources for learning . . . . .	258
<b>Chapter 12. Balancing workloads with clusters . . . . .</b>	<b>261</b>
Clusters and workload management . . . . .	261
Clusters and node groups . . . . .	262
Workload management (WLM) for distributed platforms . . . . .	263
Techniques for managing state . . . . .	263
Creating clusters . . . . .	264

Creating a cluster : Basic cluster settings . . . . .	266
Creating a cluster : Basic cluster member settings . . . . .	266
Creating a cluster : Summary settings . . . . .	267
Server cluster collection . . . . .	267
Creating cluster members . . . . .	269
Cluster member collection . . . . .	270
Creating backup clusters . . . . .	271
Backup clusters . . . . .	272
Backup cluster settings . . . . .	274
Starting clusters . . . . .	275
Stopping clusters . . . . .	276
Replicating data across application servers in a cluster . . . . .	276
Replication . . . . .	277
Replication domain collection. . . . .	278
Migrating V6.0 servers from multi-broker replication domains to data replication domains. . . . .	280
Replicating data with a multi-broker replication domain . . . . .	283
Deleting clusters . . . . .	289
Deleting cluster members . . . . .	289
Tuning a workload management configuration . . . . .	289
Workload management run-time exceptions . . . . .	290
Clustering and workload management: Resources for learning . . . . .	290
<b>Chapter 13. Setting up a high availability environment . . . . .</b>	<b>293</b>
High availability manager . . . . .	294
High availability network components. . . . .	296
Transport protocol for a high availability manager . . . . .	297
Creating a new core group . . . . .	298
Core groups . . . . .	300
High availability groups . . . . .	302
Core group collection . . . . .	303
Core group settings . . . . .	303
Changing a core group's configuration . . . . .	306
Core group and core group bridge custom properties . . . . .	306
Changing the configuration of a high availability group . . . . .	308
High availability group servers collection . . . . .	309
High availability group settings . . . . .	309
High availability group members . . . . .	310
Creating a policy for a high availability group . . . . .	311
Core group policies . . . . .	314
Core group policy settings. . . . .	315
New core group policy definition . . . . .	317
Preferred servers . . . . .	318
Preferred coordinator servers . . . . .	318
Matching criteria collection. . . . .	318
Match criteria settings . . . . .	319
Static group servers collection . . . . .	319
Changing the policy of a high availability group . . . . .	319
Adding members to a core group . . . . .	320
Core group servers . . . . .	321
Core group server settings . . . . .	321
Core group server move settings . . . . .	322
Routing high availability group work to a different server. . . . .	322
Configuring the core group bridge service . . . . .	323
Core group communications using the core group bridge service . . . . .	324
Configuring communication between core groups that are in the same cell . . . . .	329
Configuring communication between core groups that are in different cells . . . . .	334

Configuring core group communication using a proxy peer access point . . . . .	341
Core group communication . . . . .	343
Access point group collection . . . . .	343
Access point group settings . . . . .	344
<b>Notices . . . . .</b>	<b>345</b>
<b>Trademarks and service marks . . . . .</b>	<b>347</b>



---

## 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 919-254-0206.

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.



---

## Chapter 1. Overview for setting up application serving environments

This topic summarizes the contents and organization of the administration documentation, including links to conceptual overviews and descriptions of new features.

This publication is for the administrator who is responsible for integrating application serving capabilities into an existing network environment. It looks at the product as part of a larger system, typically a production environment or realistic test environment.

This publication reiterates some installation and customization activities, including topology planning and creating product configurations. It carries the focus into the administrative realm, discussing port configuration and other network concerns. See also the *Installing your application serving environment* PDF.

This information expands the topology planning discussion by describing how to set up and maintain logical administrative domains of cells and nodes, and how to balance workload through clustering and high availability configurations.

---

### Setting up WebSphere Application Server products

IBM WebSphere Application Server products provide a next-generation application server on an industry-standard foundation. Each product addresses a distinct set of scenarios and needs. WebSphere Application Server, Version 6 product offerings are described on the WebSphere Application Server Web site at <http://www.ibm.com/software/webservers/appserv/was/>.

#### Planning

See Chapter 4, “Planning the installation (diagrams),” on page 15 for a description of typical scenarios for each WebSphere Application Server product.

#### Installing

See the *Installing your application serving environment* PDF for a description of installing the WebSphere Application Server product and other installable components on the product disc.

#### Configuring

See “Using the Profile creation wizard” on page 45 for a description of installing profiles that define a deployment manager, a managed node, or a stand-alone Application Server.

#### Migrating

See the *Migrating, coexisting, and interoperating* PDF for a description of how to migrate applications and configuration data from a previous version of WebSphere Application Server.

#### Deploying applications

The Information Center describes a way to sample WebSphere Application Server functionality by quickly deploying Web components, such as servlets and JSP files. The method is not recommended as an official development method. See the *Developing and deploying applications* PDF to get started.

---

## Introduction: System administration

A variety of tools are provided for administering the WebSphere Application Server product:

- **Console**

The administrative console is a graphical interface that provides many features to guide you through deployment and systems administration tasks. Use it to explore available management options.

For more information, refer to “Introduction: Administrative console.”

- **Administrative agents**

Servers, nodes and node agents, cells and the deployment manager are fundamental concepts in the administrative universe of the product. It is also important to understand the various processes in the administrative topology and the operating environment in which they apply.

For more information, refer to “Welcome to basic administrative architecture” on page 4.

- **Scripting**

The WebSphere administrative (wsadmin) scripting program is a powerful, non-graphical command interpreter environment enabling you to run administrative operations in a scripting language. You can also submit scripting language programs to run. The wsadmin tool is intended for production environments and unattended operations.

For more information, refer to “Introduction: Administrative scripting (wsadmin)” on page 3.

- **Commands**

Command-line tools are simple programs that you run from an operating system command-line prompt to perform specific tasks, as opposed to general purpose administration. Using the tools, you can start and stop application servers, check server status, add or remove nodes, and complete similar tasks.

For more information, refer to “Introduction: Administrative commands” on page 4.

- **Programming**

The product supports a Java programming interface for developing administrative programs. All of the administrative tools supplied with the product are written according to the API, which is based on the industry standard Java Management Extensions (JMX) specification.

For more information, refer to “Introduction: Administrative programs” on page 4.

- **Data**

Product configuration data resides in XML files that are manipulated by the previously-mentioned administrative tools.

For more information, refer to “Introduction: Administrative configuration data” on page 4.

## Introduction: Administrative console

The administrative console is a graphical interface for performing deployment and system administration tasks. It runs in your Web browser. Your actions in the console modify a set of XML configuration files.

You can use the console to perform tasks such as:

- Add, delete, start, and stop application servers
- Deploy new applications to a server
- Start and stop existing applications, and modify certain configurations
- Add and delete Java 2 Platform, Enterprise Edition (J2EE) resource providers for applications that require data access, mail, URLs, and so on
- Manage variables, shared libraries, and other configurations that can span multiple application servers
- Configure product security, including access to the administrative console
- Collect data for performance and troubleshooting purposes
- Find the product version information. It is located on the front page of the console.

See the *Using the administrative clients* PDF for information on how you begin using the console. See also the **Reference > Administrator > Settings** section of the Information Center navigation. It lists the settings or properties you can configure.

## Introduction: Administrative scripting (wsadmin)

The WebSphere administrative (wsadmin) scripting program is a powerful, non-graphical command interpreter environment enabling you to run administrative operations in a scripting language. The wsadmin tool is intended for production environments and unattended operations. You can use the wsadmin tool to perform the same tasks that you can perform using the administrative console.

The following list highlights the topics and tasks available with scripting. See the *Administering applications and their environment* PDF for more information on how to perform these tasks.

- Getting started with scripting Provides an introduction to WebSphere Application Server scripting and information about using the wsadmin tool. Topics include information about the scripting languages and the scripting objects, and instructions for starting the wsadmin tool.
- Deploying applications Provides instructions for deploying and uninstalling applications. For example, stand-alone Java archive files and Web archive files, the administrative console, remote enterprise archive (EAR) files, file transfer applications, and so on.
- Managing deployed applications Includes tasks that you perform after the application is deployed. For example, starting and stopping applications, checking status, modifying listener address ports, querying application state, configuring a shared library, and so on.
- Configuring servers Provides instructions for configuring servers, such as creating a server, modifying and restarting the server, configuring the Java virtual machine, disabling a component, disabling a service, and so on.
- Configuring connections to Web servers Includes topics such as regenerating the plug-in, creating new virtual host templates, modifying virtual hosts, and so on.
- Managing servers Includes tasks that you use to manage servers. For example, stopping nodes, starting and stopping servers, querying a server state, starting a listener port, and so on.
- Clustering servers Includes topics about clusters, such as creating clusters, creating cluster members, querying a cluster state, removing clusters, and so on.
- Configuring security Includes security tasks, for example, enabling and disabling global security, enabling and disabling Java 2 security, and so on.
- Configuring data access Includes topics such as configuring a Java DataBase Connectivity (JDBC) provider, defining a data source, configuring connection pools, and so on.
- Configuring messaging Includes topics about messaging, such as Java Message Service (JMS) connection, JMS provider, WebSphere queue connection factory, MQ topics, and so on.
- Configuring mail, URLs, and resource environment entries Includes topics such as mail providers, mail sessions, protocols, resource environment providers, reference tables, URL providers, URLs, and so on.
- Dynamic caching Includes caching topics, for example, creating, viewing and modifying a cache instance.
- Troubleshooting Provides information about how to troubleshoot using scripting. For example, tracing, thread dumps, profiles, and so on.
- Obtaining product information Includes tasks such as querying the product identification.
- Scripting reference material Includes all of the reference material related to scripting. Topics include the syntax for the wsadmin tool and for the administrative command framework, explanations and examples for all of the scripting object commands, the scripting properties, and so on.

## Introduction: Administrative commands

Command-line tools are simple programs that you run from an operating system command-line prompt to perform specific tasks, as opposed to general purpose administration. Using the tools, you can start and stop application servers, check server status, add or remove nodes, and complete similar tasks.

See the *Administering applications and their environment* PDF for the names and syntax of all the commands that are available with the product. A subset of these commands are particular to system administration purposes.

## Introduction: Administrative programs

The product supports a Java programming interface for developing administrative programs. All of the administrative tools supplied with the product are written according to the API, which is based on the industry standard Java Management Extensions (JMX) specification. You can write a Java program that performs any of the administrative features of the WebSphere Application Server administrative tools. You can also extend the basic WebSphere Application Server administrative system to include your own managed resources.

## Introduction: Administrative configuration data

Administrative tasks typically involve defining new configurations of the product or performing operations on managed resources within the environment. IBM WebSphere Application Server configuration data is kept in files. Because all product configuration involves changing the content of those files, it is useful to know the structure and content of the configuration files.

The WebSphere Application Server product includes an implementation of the Java Management Extension (JMX) specification. All operations on managed resources in the product go through JMX functions. This setup means a more standard framework underlying your administrative operations as well as the ability to tap into the systems management infrastructure programmatically.

## Welcome to basic administrative architecture

This article discusses basic concepts in the administrative architecture to help you understand system administration in a WebSphere Application Server environment. The fundamental concepts for WebSphere Application Server administration include software processes called servers, topological units referenced as nodes and cells, and the configuration repository used for storing configuration information.

Servers perform the actual running of the code. Several types of servers exist depending on the configuration. Each server runs in its own Java virtual machine (JVM). The application server is the primary run-time component in all WebSphere Application Server configurations. All WebSphere Application Server configurations can have one or more application servers. In some configurations, each application server functions as a separate entity. No workload distribution or common administration among application servers exists. In other configurations, workload can be distributed between servers and administration can be done from a central point.

A node is a logical group of WebSphere Application Server-managed server processes that share a common configuration repository. A node is associated with a single WebSphere Application Server profile. A WebSphere Application Server node does not necessarily have a one-to-one association with a system. One computer can host arbitrarily many nodes, but a node cannot span multiple computer systems. A node can contain zero or more application servers.

The configuration repository holds copies of the individual component configuration documents that define the configuration of a WebSphere Application Server environment. All configuration information is stored in .xml files.

A cell is a grouping of nodes into a single administrative domain. A cell can consist of multiple nodes, all administered from a deployment manager server. When a node becomes part of a cell (a federated node),

a node agent server is installed on the node to work with the deployment manager server to manage the WebSphere Application Server environment on that node.

When a node is a standalone node, not part of a cell, the configuration repository is fully contained on the node. When a node is part of a cell, the configuration and application files for all nodes in the cell are centralized into a cell master configuration repository. This centralized repository is managed by the deployment manager server and synchronized to local copies that are held on each node. The local copy of the repository that is given to each node contains just the configuration information needed by that node, not the full configuration that is maintained by the deployment manager.

### WebSphere Application Server types

This section discusses the three server types that interact to perform system administration.

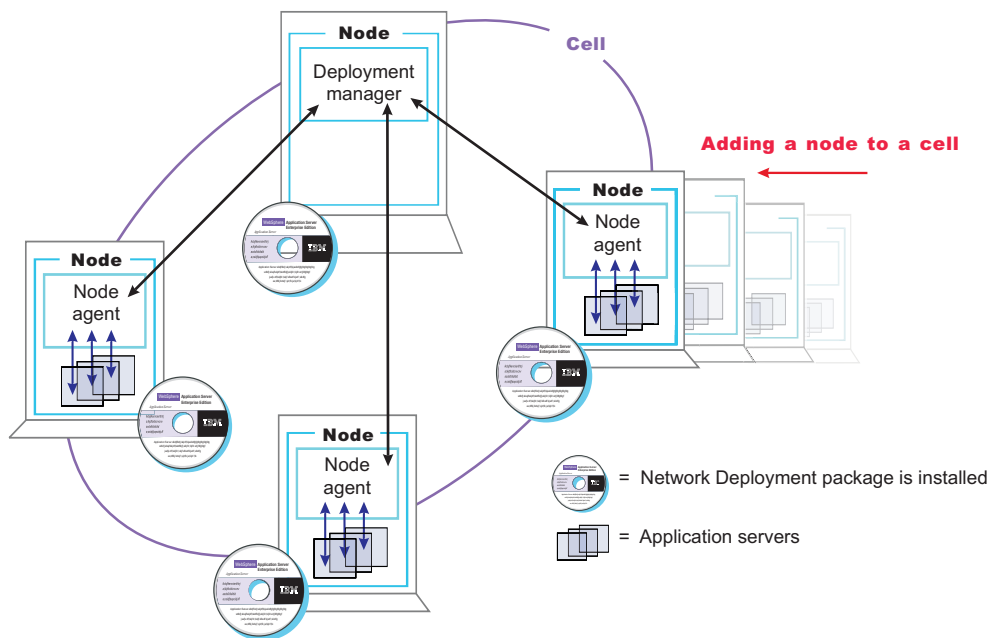
**Application Server:** A WebSphere Application Server provides the functions that are required to support and host user applications. An application server runs on only one node, but one node can support many application servers.

**Node agent:** When a node is federated, a node agent is created and installed on that node. The node agent works with the deployment manager to perform administrative activities on the node.

**Deployment manager:** With the deployment manager, you can administer multiple application servers from one centralized manager. The deployment manager works with the node agent on each node to manage all the servers in a distributed topology.

The following diagram depicts the concepts that are discussed in this article.

**IBM WebSphere Application Server Network Deployment package**



The concepts that are discussed in this article form the basis of WebSphere Application Server administration. More detailed descriptions can be found in other sections.

---

## Introduction: Servers

### Application servers

Application servers provide the core functionality of the WebSphere Application Server product family. They extend the ability of a Web server to handle Web application requests, and much more. An application server enables a server to generate a dynamic, customized response to a client request.

For additional overview, refer to “Introduction: Application servers.”

## Clusters

*Workload management* optimizes the distribution of client processing tasks. Incoming work requests are distributed to the application servers that can most effectively process the requests. Workload management also provides failover when servers are not available, improving application availability.

*Clusters* are sets of application servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines, as opposed to the servers that are part of the same node and must be located on the same host machine.

For additional overview, refer to “Introduction: Clusters” on page 8.

## Introduction: Application servers

### Overview

An application server is a Java Virtual Machine (JVM) that is running user applications. The application server collaborates with the Web server to return a dynamic, customized response to a client request. Application code, including servlets, JavaServer Pages (JSP) files, enterprise beans and their supporting classes, runs in an application server. Conforming to the Java 2 platform, Enterprise Edition (J2EE) component architecture, servlets and JSP files run in a Web container, and enterprise beans run in an Enterprise JavaBeans (EJB) container.

To begin creating and managing an application server, see Chapter 11, “Administering application servers,” on page 191.

You can define multiple application servers, each running its own JVM. Enhance the operation of an application server by using the following options:

- Configure transport chains to provide networking services to such functions as the service integration bus component of IBM service integration technologies, WebSphere Secure Caching Proxy, and the high availability manager core group bridge service. See “Configuring transport chains” on page 213 for more information.
- Plug into an application server to define a hook point that runs when the server starts and shuts down. See “Custom services” on page 228 for more information.
- Define command-line information that passes to a server when it starts or initializes. See the *Using the administrative clients* PDF for more information.
- “Tuning application servers” on page 256
- Enhance the performance of the application server JVM. See “Using the JVM” on page 244 for more information.
- Use an Object Request Broker (ORB) for RMI/IIOP communication. See the *Developing and deploying applications* PDF for more information.

### Asynchronous messaging

The product supports asynchronous messaging based on the Java Messaging Service (JMS) of a JMS provider that conforms to the JMS specification version 1.1.

The JMS functions of the default messaging provider in WebSphere Application Server are served by one or more messaging engines (in a service integration bus) that runs within application servers.



In a deployment manager cell, there can be WebSphere Application Server version 5 nodes. If a version 5 node is configured to use V5 Default Messaging (the version 5 Embedded Messaging), there can be at most one JMS server on that node.

## Generic Servers

In distributed platforms, the Generic Servers feature allows you create a generic server as an application server instance within the WebSphere Application Server administration, and associate it with a non-WebSphere server or process. The generic server can be associated with any server or process necessary to support the application server environment, including:

- A Java server
- A C or C++ server or process
- A CORBA server
- A Remote Method Invocation (RMI) server

After you define a generic server, you can use the Application Server administrative console to start, stop, and monitor the associated non-WebSphere server or process when stopping or starting the applications that rely on them.

For more information, refer to “Creating generic servers” on page 211.

## Introduction: Web servers

In the WebSphere Application Server product, an application server works with a Web server to handle requests for dynamic content, such as servlets, from Web applications. Go to <http://www.ibm.com/software/webservers/appserv/doc/latest/prereq.html> for the most current information about supported Web servers.

The application server and Web server communicate using “Web server plug-ins” on page 122. Chapter 7, “Communicating with Web servers,” on page 107 describes how to set up your Web server and Web server plug-in environment and how to create a Web server definition. The Web server definition associates a Web server with a previously defined managed or unmanaged node. After you define the Web server to a node, you can use the administrative console to perform the following functions for that Web server.

If the Web server is defined to a managed node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.
- Propagate the plug-in configuration file after it is generated.

If the Web server is an IBM HTTP Server (IHS) and the IHS Administration server is installed and properly configured, you can also:

- Display the IBM HTTP Server Error log (error.log) and Access log (access.log) files.
- Start and stop the server.
- Display and edit the IBM HTTP Server configuration file (httpd.conf).

If the Web server it is defined to an unmanaged node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.

If the Web server is an IBM HTTP Server (IHS) and the IHS Administration server is installed and properly configured, you can also:

- Display the IBM HTTP Server Error log (error.log) and Access log (access.log) files.
- Start and stop the server.

- Display and edit the IBM HTTP Server configuration file (httpd.conf).
- Propagate the plug-in configuration file after it is generated.

You can not propagate an updated plug-in configuration file to a non-IHS Web server that is defined to an unmanaged node. You must manually install an updated plug-in configuration file to a Web server that is defined to an unmanaged node. Web servers defined to an unmanaged node are typically remote Web servers. Remote Web servers are Web servers that are not located on the same machine as the WebSphere Application Server.

After you set up your Web server and Web server plug-in, whenever you deploy a Web application, you must specify a Web server as the deployment target that serves as a router for requests to the Web application. The configuration settings in the plug-in configuration file (plugin-cfg.xml) for each Web server are based on the applications that are routed through that Web server. If the Web server plug-in configuration service is enabled, a Web server plug-in's configuration file is automatically regenerated whenever a new application is associated with that Web server.

**Note:** Before starting the Web server, make sure you are authorized to run any Application Response Measurement (ARM) agent associated with that Web server.

Refer to your Web server documentation for information on how to administer that Web server. For tips on tuning your Web server plug-in, see “Web server plug-in tuning tips” on page 128.

## Introduction: Clusters

Clusters are groups of servers that are managed together and participate in workload management. A cluster can contain nodes or individual application servers. A node is usually a physical computer system with a distinct host IP address that is running one or more application servers. Clusters can be grouped under the configuration of a cell, which logically associates many servers and clusters with different configurations and applications with one another depending on the discretion of the administrator and what makes sense in their organizational environments.

Clusters are responsible for balancing workload among servers. Servers that are a part of a cluster are called cluster *members*. When you install an application on a cluster, the application is automatically installed on each cluster member.

Because each cluster member contains the same applications, you can distribute client tasks in distributed platforms according to the capacities of the different machines by assigning weights to each server.

In distributed platforms, assigning weights to the servers in a cluster improves performance and failover. Tasks are assigned to servers that have the capacity to perform the task operations. If one server is unavailable to perform the task, it is assigned to another cluster member. This reassignment capability has obvious advantages over running a single application server that can become overloaded if too many requests are made.

Node groups bound clusters. All cluster members of a given cluster must be members of the same node group. For more information about clusters and node groups, see “Clusters and node groups” on page 262.

To learn more about clusters, see “Clusters and workload management” on page 261 and Chapter 12, “Balancing workloads with clusters,” on page 261 for more information.

### Core groups

A group of clusters can be defined as a *core group*. All of the application servers defined as a member of one of the clusters included in a core group are automatically members of that core group. Individual application servers that are not members of a cluster can also be defined as a member of a core group.

The use of core groups enables WebSphere Application Server to provide high availability for applications that must always be available to end users. You can also configure core groups to communicate with each other using the *core group bridge*. The core groups can communicate within the same cell or across cells.

To learn more about core groups, see Chapter 13, “Setting up a high availability environment,” on page 293.

---

## Introduction: Environment

The environment of the product applies to the configuring of Web server plug-ins, variables, and objects that you want consistent throughout a cell.

### Web servers

In the WebSphere Application Server product, an application server works with a Web server to handle requests for Web applications. The application Server and Web server communicate using a WebSphere HTTP plug-in for the Web server.

For more information, refer to “Introduction: Web servers” on page 7.

### Cell-wide settings

Cell-wide settings are sets of configuration data that are stored in files in the cell directory. These configuration files are replicated to every node in the cell. Several different configuration settings apply to the entire cell. These settings include the definition of virtual hosts, shared libraries, and any variables that must be consistent throughout the entire cell.

For more information, refer to “Introduction: Cell-wide settings.”

### Variables

A variable is a configuration property that can be used to provide a parameter for any value in the system. A variable has a name and a value to use in place of that name wherever the variable name is located within the system.

For more information, refer to “Configuring WebSphere variables” on page 172.

## Introduction: Cell-wide settings

The configuration data for WebSphere Application Server is stored in XML files. The XML files exist in one of several directories in the configuration repository tree.

The directory in which a configuration file exists determines its scope, or how broadly or narrowly that data applies. Files in an individual server directory apply to that specific server only. Files in a node-level directory apply to every server on that node. Files in the cell directory apply to every server on every node within the entire cell.

*Cell-wide settings* are configuration files in the cell directory. The files are replicated to every node in the cell. Several different configuration settings apply to the entire cell. These settings include the definition of virtual hosts, shared libraries, and any variables that you want consistent throughout the entire cell.



---

## Chapter 2. How do I administer applications and their environments?

- Establish the application serving environment
- Secure the application serving environment. (See the *Securing applications and their environment* PDF.)
- Set up Web access for applications. (See the *Developing and deploying applications* PDF.)
- Set up resources for applications to use. (See the *Developing and deploying applications* PDF.)
- Configure class loaders - see development and deployment. (See the *Developing and deploying applications* PDF.)
- Deploy and administer applications. (See the *Developing and deploying applications* PDF.)
- Use the administrative clients. (See the *Using administrative clients* PDF.)
- Troubleshoot deployment and administration. (See the *Troubleshooting and support* PDF.)

### Establish the application serving environment

The following tasks involve establishing application serving capability in your network environment, whether you use single or clustered application servers. Servers can be grouped into administrative domains known as nodes and cells.

See also the overview:

- Version 6 topology and terminology

---

### Create WebSphere profiles

Profiles are the files that define a stand-alone Application Server node, a managed node, or a deployment manager node. A profile also includes all of the files that the node can change.

---

### Administer nodes

A node is a grouping of managed servers. Use this task to view information about and manage nodes.

---

### Administer 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. A node agent is created automatically when a node is added to a cell.

---

### Administer cells

When you installed the WebSphere Application Server Network Deployment product, a cell was created. A cell provides a way to group one or more nodes of your Network Deployment product. You probably will not need to reconfigure the cell. Use this task to view information about and manage a cell.

---

### Administer configurations

Application server configuration files define the available application servers, their configurations, and their contents. You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

---

### **Configure 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. 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.

---

### **Administer application servers**

Create, configure, and operate application server processes. An application server configuration provides settings that control how an application server provides services for running enterprise applications and their components.

---

### **Administer other server types**

One step in the process of creating an application server is to specify a template. A server template is used to define the configuration settings of the new server. You have the option of specifying the default server template or choosing a template that is based on a server that already exists. The default template will be used if you do not specify a different template when you create the server.

You can create other types of servers, to represent Web servers in your topology, or for other purposes. There are two types of *generic* servers: (1) Non-Java applications or processes, or (2) Java applications or processes. A *custom service* provides the ability to plug into a WebSphere application server to define a hook point that runs when the server starts and shuts down.

---

### **Balance workloads by clustering application servers**

To monitor application servers and manage the workloads of servers, use server clusters and cluster members provided by the Network Deployment product.

---

### **Establishing high availability (HA) for failover**

Planning ahead for high availability support is important in order to avoid the risk of a failure without failover coverage. The application server runtime of the infrastructure managed by a high availability manager includes such entities as cells and clusters. These components relate closely to core groups, high availability groups, and the policy that defines the high availability infrastructure. In a properly configured high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment.

---

---

## Chapter 3. Setting up the application serving environment

This topic summarizes the contents of the documentation that helps you set up your application serving environment. This information is for administrators, particularly those performing installation, customization, and maintenance of topologies.

### Chapter 4, “Planning the installation (diagrams),” on page 15

In preparation for installation, this topic describes common product topologies that you can install with WebSphere Application Server, Version 6 products.

### Chapter 5, “Configuring the product after installation,” on page 39

This topic describes what to do after installing the product.

If you are using Network Deployment, you must configure a profile.

You can display the First Steps tool, an easy way to get started with the product.

### Chapter 6, “Configuring ports,” on page 103

This topic provides information about port number settings for Version 6 and previous versions, for use in coexistence and interoperability situations.

### Chapter 7, “Communicating with Web servers,” on page 107

This topic describes how to install and configure WebSphere plug-ins for Web servers, enabling communication between Web servers and application servers.

### Chapter 8, “Setting up the administrative architecture,” on page 131

This topic describes how to set up logical administrative domains, including cells and nodes.

### Chapter 9, “Configuring the environment,” on page 165

This topic describes how to configure cell-wide settings for virtual hosts, variables and shared libraries to assist in handling requests among Web applications, Web containers, and application servers.

### Chapter 10, “Working with server configuration files,” on page 183

This topic describes how to change the default locations of configuration files, as needed.

Application server configuration files define the available application servers, their configurations, and their contents.

### Chapter 11, “Administering application servers,” on page 191

This topic describes how to configure individual application servers to provide services for running enterprise applications and their components.

### Chapter 12, “Balancing workloads with clusters,” on page 261

This topic describes how to configure clusters, which are sets of servers that are managed together and participate in workload management.

### Chapter 13, “Setting up a high availability environment,” on page 293

This topic describes planning ahead for high availability support, which is important in order to avoid the risk of a failure without failover coverage. In a properly setup high availability environment, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment.





---

## Chapter 4. Planning the installation (diagrams)

This topic describes common product topologies that you can install with the product.

Use this topic to understand the capabilities of your product package. Knowing what you can do with the product might influence how you install the product and other installable components on the product disc.

This topic describes topology diagrams and shows you how to create the topologies by showing what components to install for each topology.

### Phased installation roadmap

To install a Version 6 production environment, you install the following components:

- The WebSphere Application Server product on your product CD
- A supported Web server, such as the IBM HTTP Server V6 on the product CD
- A binary plug-in module for your Web server from the product CD

You can also use the product CD to install an application client environment on a client machine. Running Java 2 Platform, Enterprise Edition (J2EE) and thin application clients that communicate with WebSphere Application Server requires that elements of the Application Server are installed on the machine on which the client runs. However, if the machine does not have the Application Server installed, you can install Application Server clients to provide a stand-alone client run-time environment for your client applications.

You can use the Application Server Toolkit CD in the primary packet of discs to install a development environment. Or you can use the Rational Application Developer Trial CD in the supplemental packet of discs to install a fully integrated development environment that includes an exact replica of the Application Server for development testing.

### Installation features

Installation features in V6 include:

Feature	Description
The Network Deployment product includes Application Server and managed nodes.	You can use the Profile creation wizard to create deployment manager profiles, Application Server profiles, and custom profiles after installing the WebSphere Application Server Network Deployment product. You do not have to install another set of binary files to install an Application Server. All profiles on a machine share the core product files of the Profile creation wizard. (If you install again to create another set of core product files, there are two Profile creation wizards. One for each set of core product files.)
The product CD includes all of the installable components that are required to create an e-business environment.	You can use the product CD to install the IBM HTTP Server, the Web server plug-ins, and the WebSphere Application Server Clients. You do not have to use separate CDs. Separate installation programs exist within component directories on the product CD.
Each installable component has its own installation program.	You can use the V6 launchpad to install any installable component on the product CD. Or you can install each component directly using the <b>install</b> command in each component directory.
The launchpad can install any installable product in the primary packet of compact discs.	The launchpad can also install the Application Server Toolkit on Windows 2000 and Linux (Intel) systems. The Application Server Toolkit is on a separate disc, which requires you to change discs to launch the installation.

Review topology diagrams for each of the following installable components to determine which topology best fits your needs. The diagrams and their accompanying procedures can serve as a roadmap for installing a similar topology.

This topic describes installation scenarios for the following installable components:

- WebSphere Application Server Network Deployment
- Web server plug-ins
- Application clients

In addition to product installation diagrams for the installable components, this topic also links to a roadmap for using the Profile creation wizard, which is new for Version 6. The Profile creation wizard lets you create run-time environments for application server processes.

Each of the following installation scenarios includes topology diagrams and associated installation steps. Each step links to a specific procedure for installing a component or to a description of a command or tool.

1. Review the installation scenarios for the Network Deployment product, as described in “Planning to install Network Deployment.”
2. Review the installation scenarios for the WebSphere Application Server plug-ins, as described in “Planning to install Web server plug-ins” on page 25.
3. Review the installation scenarios for the application clients, as described in “Planning to install WebSphere Application Server Clients” on page 31.
4. Review the installation scenarios for the Profile creation wizard, as described in “Planning to create application server environments” on page 32.
5. **Optional:** Review interoperability and coexistence diagrams to know what is possible with Version 6. WebSphere Application Server V6 can interoperate with your other e-business systems, including other versions of WebSphere Application Server. *Interoperability* provides a communication mechanism for WebSphere Application Server nodes that are at different versions. *Coexistence* describes multiple versions or instances running on the same machine, at the same time.

Interoperability support enhances migration scenarios with more configuration options. It often is convenient or practical to interoperate during the migration of a configuration from an earlier WebSphere Application Server version to a later one when some machines are at the earlier version and some machines are at the later version. The mixed environment of machines and application components at different software version levels requires interoperability and coexistence.

It is often impractical, or even physically impossible, to migrate all the machines and applications within an enterprise at the same time. Understanding multiversion interoperability and coexistence is therefore an essential part of a migration between version levels.

See the *Migrating, coexisting, and Interoperating* PDF for more information.

6. **Optional:** Consider performance when designing your network, as described in “Example: Choosing a topology for better performance” on page 33 and “Queuing network” on page 34.

You can review installation scenarios to identify the specific steps to follow when installing more than one component on a single machine or on separate machines.

After determining an appropriate installation scenario, you are ready to install the necessary components and to configure the products for the system that you selected.

---

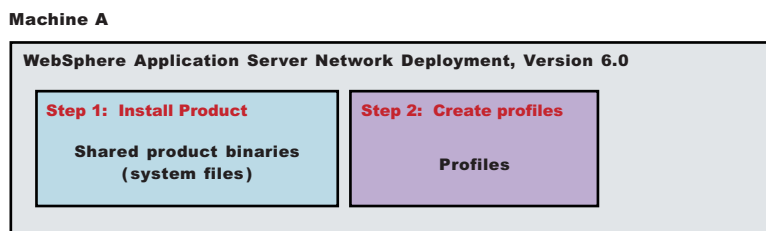
## Planning to install Network Deployment

This topic describes common installation scenarios and links to component installation procedures.

In Version 6.0, installing WebSphere Application Server Network Deployment is a two-step process. The first step is using the installation wizard to install a shared set of core product files. The second step is using the *Profile creation wizard* to create a *deployment manager profile*, an *Application server profile*, or a *custom profile*.

A profile is a separate data partition that includes the files that define a run-time environment for an application server process.

A running application server process, such as a deployment manager, can create, read, update, or delete the configuration files, data files, and log files in its profile. The application server process has read-only access to the system files, which include command files and other shared product binary files. System files are updated only by installing refresh packs or fix packs, or by products that extend WebSphere Application Server Network Deployment.



## Scenarios for installation

The following information describes scenarios for installing the product in various topologies on one or more machines. Two types of WebSphere Application Server topologies are possible using the Network Deployment product:

- Topologies for a stand-alone application server
- Topologies for a managed group of application servers

### Topologies for a stand-alone application server

Each stand-alone application server has its own administrative console and runs independently of other application servers.

The following topologies are described in this topic.

- **Scenario 1:** Single-machine installation of a stand-alone application server
- **Scenario 2:** Single-machine installation of a stand-alone application server and a Web server
- **Scenario 3:** Two-machine installation of a stand-alone application server and a Web server
- **Scenario 4:** Two-machine installation of multiple stand-alone application servers and a Web server

### Topologies for a managed group of application servers

A managed group of application servers is called a *cell*. A cell consists of one deployment manager and one or more federated application servers, which are called *managed nodes*.

A node becomes a managed node in either of two ways:

- Federating the node within an *Application Server profile* into the cell
- Federating the node within a *custom profile* into the cell

The deployment manager is the single point of administration for all of the managed nodes in the cell. The deployment manager maintains the configuration files for nodes that it manages and deploys applications to those managed nodes.

The following topologies for a cell are described in this topic.

- **Scenario 5:** Single-machine installation of a cell of application servers
- **Scenario 6:** Single-machine installation of a cell of application servers and a Web server
- **Scenario 7:** Two-machine installation of a cell of application servers and a Web server
- **Scenario 8:** Three-machine installation of a cell of application servers and a Web server

Each of the following scenarios includes a diagram and a list of detailed installation steps.

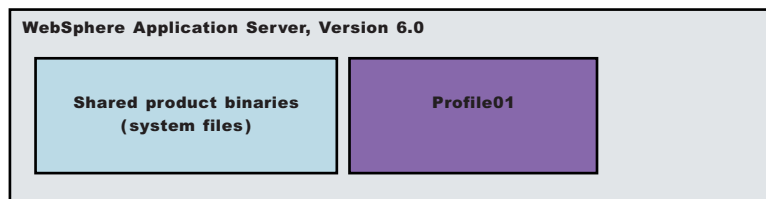
Some scenarios are more typical in production environments. For example, Scenario 1 supports a lighter workload than Scenario 3 or Scenario 4. However, Scenario 1 is a fully functional environment. Scenarios 3 and 4 are typical production environments for a stand-alone application server. Scenario 8 is a typical production scenario for a cell environment.

- **Scenario 1:** Install a stand-alone application server on a single machine.

Installing WebSphere Application Server Network Deployment by itself on a single machine lets you create a stand-alone Application Server profile. Each stand-alone application server profile includes a server1 application server process. Installing Network Deployment creates the set of system files. The Profile creation wizard creates the profile for the application server. The profile is a separate data partition with files that define the stand-alone application server environment.

In this scenario, the application server uses its internal HTTP transport chain for communication, which is suitable for handling an application with a relatively low request work load. For example, this type of installation can support a simple test environment or a departmental intranet environment.

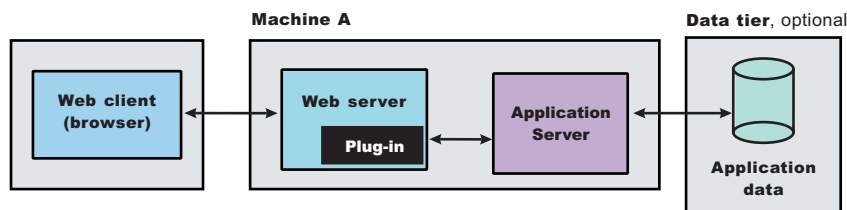
**Machine A**



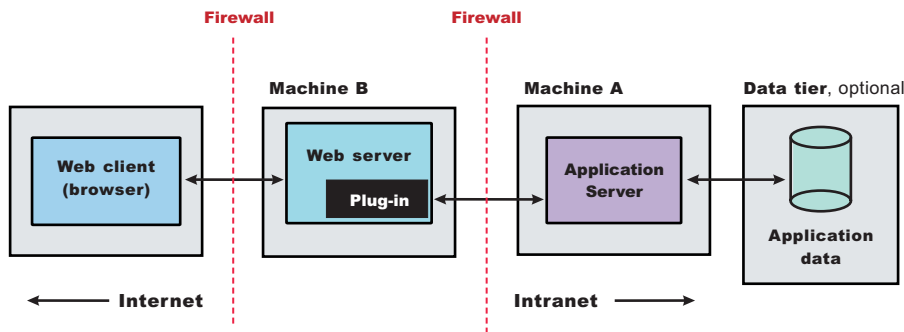
1. Install WebSphere Application Server Network Deployment.
2. Create an Application Server profile using the Profile creation wizard.

- **Scenario 2:** Install a stand-alone application server and a Web server on a single machine.

Installing a Web server, such as IBM HTTP Server, on the same machine as the application server provides a more robust Web server environment. Installing a Web server plug-in is a requirement for the Web server to communicate with the application server. This type of installation supports rigorous testing environments or production environments that do not require a firewall. However, this is not a typical production environment.



1. Install WebSphere Application Server Network Deployment.
  2. Create an Application Server profile using the Profile creation wizard.
  3. Install IBM HTTP Server or another supported Web server.
  4. Install the Web server plug-ins and configure the Web server using the Plug-ins installation wizard.
- **Scenario 3:** Install a stand-alone application server and a Web server on separate machines.  
In the typical production environment, the application server on one machine communicates with a Web server on a separate (remote) machine through the Web server plug-in. Optional firewalls can provide additional security for the application server machine.



1. Install WebSphere Application Server Network Deployment on Machine A.
  2. Create an Application Server profile using the Profile creation wizard on Machine A.
  3. Install IBM HTTP Server or another supported Web server on Machine B.
  4. Install the Web server plug-ins and configure the Web server using the Plug-ins installation wizard.
  5. The Plug-ins installation wizard creates a script named `configureWeb_server_name` in the `plugins_install_root/bin` directory on Machine B. Copy the script to the `install_root/bin` directory on Machine A.
  6. Run the `configureWeb_server_name` script to create a Web server definition in the administrative console. You can then use the administrative console to manage the Web server.
  7. Propagate the `plugin-cfg.xml` file from the application server to the Web server using the administrative console. Click **Servers > Web server > Propagate Plug-in**. (Web servers other than IBM HTTP Server require manual propagation.)
- **Scenario 4:** Install multiple stand-alone application servers on one machine and a Web server on a separate machine.

The Profile creation wizard can create a deployment manager profile, an application server profile, or a custom profile. Each profile is a separate data partition containing the files that define the run-time environment. After creating a profile and installing a dedicated Web server, use the Plug-ins installation wizard to install a plug-in and to update the Web server configuration file. The Web server can then communicate with the application server.

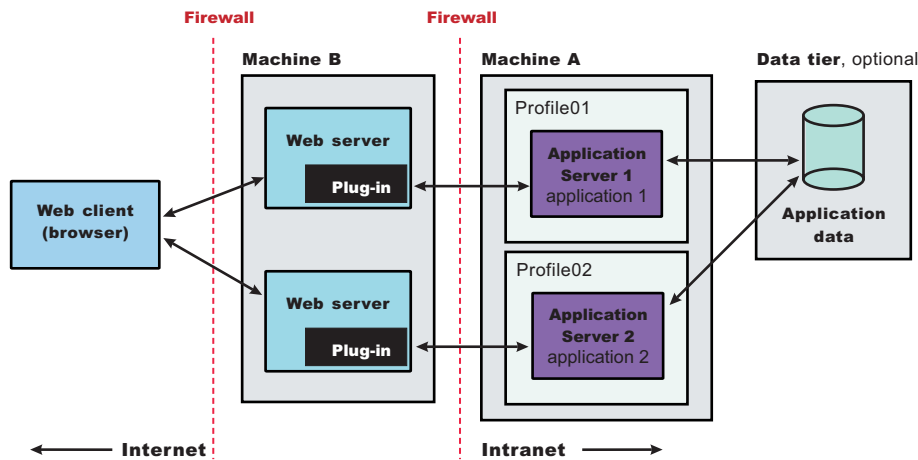
This topology lets each profile have unique applications, configuration settings, data, and log files, while sharing the same set of system files. Creating multiple profiles creates multiple application server environments that you can dedicate to different purposes.

For example, each application server on a Web site can serve a different application. In another example, each application server can be a separate test environment that you assign to a programmer or a development team.

### Updating the core product files

Another feature of having multiple profiles is enhanced serviceability. When a refresh pack or a fix pack updates the core product files on a machine, all of the application server profiles that were created from the core product files begin using the updated files. In some situations, you might prefer to not update

all of the application servers on a machine. In such situations, simply install the product a second time to create a second set of core product files. Create application server profiles from both installations to manage the product updates incrementally.



1. Install WebSphere Application Server Network Deployment on Machine A.
2. Create the first Application Server profile using the Profile creation wizard on Machine A.
3. Install IBM HTTP Server or another supported Web server on Machine B.
4. On Machine B, install the Web server plug-ins and configure the first Web server using the Plug-ins installation wizard.
5. The Plug-ins installation wizard creates a script named `configureWeb_server_name` in the `plugins_install_root/bin` directory on Machine B. Copy the script to the `install_root/bin` directory on Machine A.
6. Run the `configureWeb_server_name` script to create a Web server definition in the administrative console. You can then use the administrative console to manage the Web server.
7. Propagate the `plugin-cfg.xml` file from the application server to the Web server using the administrative console. Click **Servers > Web server > Propagate Plug-in**. (Web servers other than IBM HTTP Server require manual propagation.)
8. Create the second Application Server profile using the Profile creation wizard on Machine A. Make the profile the default profile during the profile creation by selecting the check box on the appropriate panel.  
The script that the Plug-ins installation wizard creates works on the default profile only. So, this script can only create a Web server definition on the profile that is the default profile at the time that the script runs.
9. Install a second IBM HTTP Server or another supported Web server on Machine B.
10. On Machine B, install the Web server plug-ins to configure the second Web server using the Plug-ins installation wizard. Both Web servers share a single installation of the plug-in binaries but must be configured individually.
11. The Plug-ins installation wizard creates a script named `configureWeb_server_name` for the second Web server. The script is in the `plugins_install_root/bin` directory on Machine B. Copy the script to the `install_root/bin` directory on Machine A.
12. Run the `configureWeb_server_name` script to create a Web server definition in the administrative console. You can then use the administrative console to manage the Web server.
13. Propagate the `plugin-cfg.xml` file from the second application server to the Web server using the administrative console. Click **Servers > Web server > Propagate Plug-in**. (Web servers other than IBM HTTP Server require manual propagation.)

- **Scenario 5:** Install a cell of managed application server nodes on one machine.

WebSphere Application Server Network Deployment can create a cell of managed application servers on a single machine from one installation of the core product files. The Profile creation wizard creates the deployment manager. After starting the deployment manager, return to the Profile creation wizard to create one or more application servers for the cell. Application server profiles have a default application server, called `server1`, and default applications. An Application Server node becomes a managed node after federating the node into the deployment manager cell.

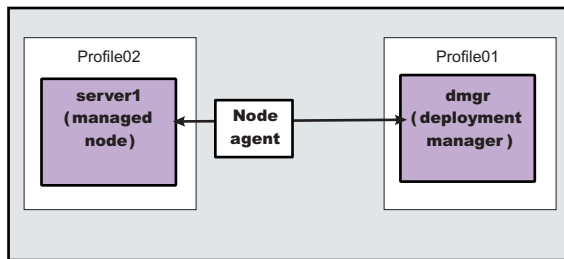
The deployment manager provides the administration for all managed nodes that are in its cell. Periodically the configuration and application files on a managed node refresh from the master copy of the files hosted on the deployment manager during *synchronization*.

In certain secure environments, the Profile creation wizard cannot federate a custom profile into a cell. Such cases require you to use the **addNode** command instead. If you have configured the deployment manager to use a JMX connector type other than the default SOAP connector, use the **addNode** command to add the node to the cell.

The deployment manager provides the administration for all managed nodes that are in its cell. Periodically the deployment manager refreshes the configuration files and application files on the managed node. Copying the master version of the files hosted on the deployment manager to the managed nodes is a process called *synchronization*.

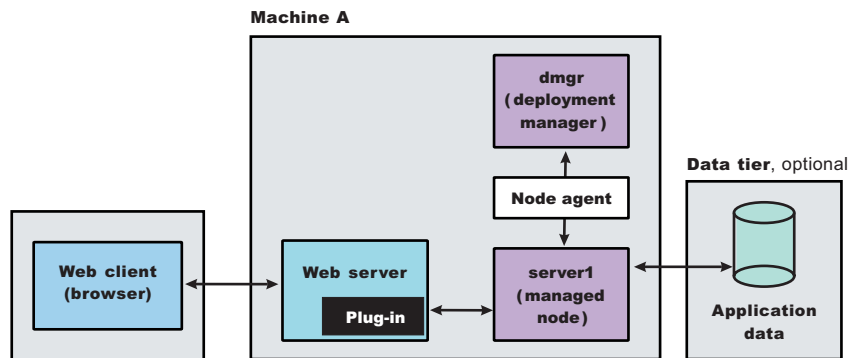
In a cell environment, only the managed nodes serve applications, not the deployment manager. The managed node in this scenario uses its internal HTTP transport chain for communication, which is suitable for an application with a relatively low request work load. For example, this type of installation can support a simple test environment or a departmental intranet environment.

**Machine A**



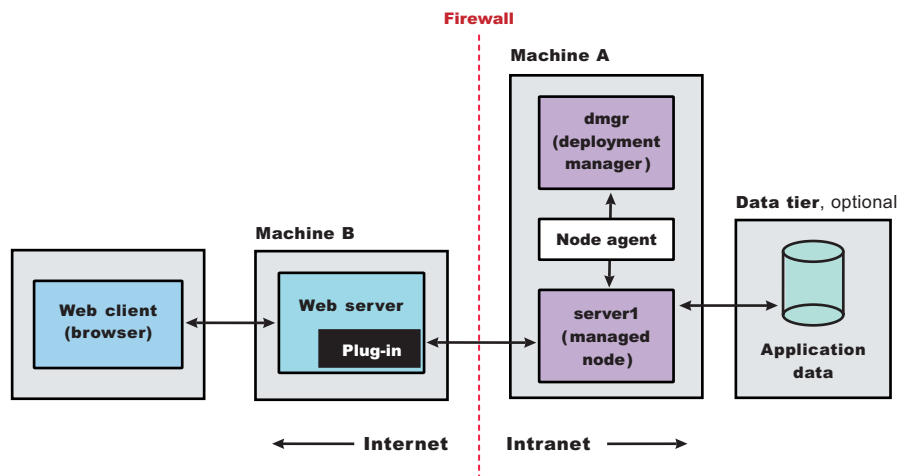
1. Install WebSphere Application Server Network Deployment.
  2. Create a deployment manager profile using the Profile creation wizard.
  3. Start the deployment manager using the First steps console or the **startManager** command.
  4. Create an Application Server profile using the Profile creation wizard.
  5. Start the application server using the First steps console or the **startServer server1** command.
  6. Add the application server node to the cell using the administrative console of the deployment manager. Click **System Administration > Nodes** to add the node.
- **Scenario 6:** Install a cell of managed application server nodes and a Web server on one machine. Installing a Web server, such as IBM HTTP Server, on the same machine as the application server provides a richer set of configuration options. Installing a Web server plug-in is required for the Web server to communicate with the server in the managed node. This type of installation can support either rigorous testing in a cell environment or production environments that do not require a firewall.





1. Install WebSphere Application Server Network Deployment.
  2. Create a deployment manager profile using the Profile creation wizard.
  3. Start the deployment manager using the First steps console or the **startManager** command.
  4. Create an Application Server profile using the Profile creation wizard.
  5. Start the application server using the First steps console or the **startServer server1** command.
  6. Add the application server node to the cell using the administrative console of the deployment manager. Click **System Administration > Nodes** to add the node.
  7. Install IBM HTTP Server or another supported Web server.
  8. Install the Web server plug-ins and configure the Web server using the Plug-ins installation wizard.
  9. The Plug-ins installation wizard creates a script named `configureWeb_server_name` in the `plugins_install_root/bin` directory. Run the `configureWeb_server_name` script to create a Web server definition in the administrative console. You can then use the administrative console to manage the Web server.
- **Scenario 7:** Install a cell of managed application server nodes on one machine and a Web server on a separate machine.

In a typical production environment, a managed node in a cell communicates with a Web server on a separate (remote) machine through the Web server plug-in. An optional firewall can provide additional security for the application server machine.



1. Install WebSphere Application Server Network Deployment on Machine A.
2. Create a deployment manager profile using the Profile creation wizard on Machine A.
3. Start the deployment manager using the First steps console or the **startManager** command on Machine A.

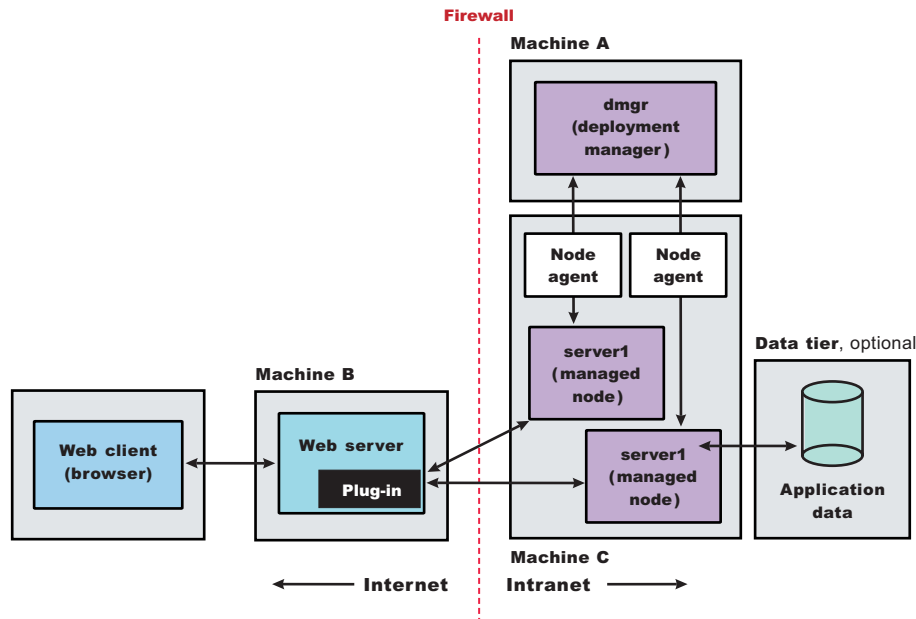


4. Create the Application Server profile and make this profile the default profile using the Profile creation wizard on Machine A.
  5. Start the application server using the First steps console or the **startServer server1** command on Machine A.
  6. Add the application server node to the cell using the administrative console of the deployment manager on Machine A. Click **System Administration > Nodes** to add the node.
  7. Install IBM HTTP Server or another supported Web server on Machine B.
  8. On Machine B, install the Web server plug-ins and configure the Web server using the Plug-ins installation wizard.
  9. The Plug-ins installation wizard creates a script named `configureWeb_server_name` in the `plugins_install_root/bin` directory on Machine B. Copy the script from Machine B to the `install_root/bin` directory on Machine A.  
 You have the option of using the script to create the Web server definition in the configuration of the deployment manager or using the administrative console of the deployment manager to create the Web server definition.
  10. Run the `configureWeb_server_name` script on Machine A to create a Web server definition or use the administrative console of the deployment manager to create the definition. You can then use the administrative console to manage the Web server.
  11. Propagate the `plugin-cfg.xml` file from the deployment manager on Machine A to the Web server on Machine B using the administrative console. Click **Servers > Web server > Propagate Plug-in**. (Web servers other than IBM HTTP Server require manual propagation.)
- **Scenario 8:** Install a deployment manager on one machine, multiple managed application server nodes on a second machine, and a Web server on a third machine.

The primary advantage of a cell over a stand-alone application server is its scalability. Managing a cell to keep it in proportion with workload levels is possible. In this scenario, managed nodes exist on Machine C. All of the managed nodes are federated into the same deployment manager. Depending on your needs, an application server in each managed node could serve the same or different applications. Having multiple machines and multiple application server profiles lets you use vertical and horizontal scaling:

- *Vertical scaling* creates multiple managed nodes on the same physical machine.
- *Horizontal scaling* creates cell members on multiple physical machines.

The managed nodes in this scenario communicate with the same Web server. However, the preferred strategy is to have a dedicated Web server for each managed node.



1. Install WebSphere Application Server Network Deployment on Machine A.
2. Create a deployment manager profile using the Profile creation wizard on Machine A.
3. Start the deployment manager using the First steps console or the **startManager** command on Machine A.
4. Install WebSphere Application Server Network Deployment on Machine C.
5. Create the first Application Server profile using the Profile creation wizard on Machine C.
6. Start the first application server using the First steps console or the **startServer server1** command on Machine C.
7. Create the second Application Server profile and make this profile the default profile using the Profile creation wizard on Machine C.
8. Start the second Application Server using the First steps console or the **startServer server1** command on Machine C.
9. Add both application server nodes to the cell using the administrative console of the deployment manager on Machine A. Click **System Administration > Nodes** to add the nodes.
10. [http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/aes/ae/tihs\\_install.htm](http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/aes/ae/tihs_install.htm) or another supported Web server on Machine B.
11. Install the Web server plug-ins and configure the Web server using the Plug-ins installation wizard on Machine B.
12. The Plug-ins installation wizard creates a script named `configureWeb_server_name` in the `plugins_install_root/bin` directory on Machine B. Copy the script from Machine B to the `install_root/bin` directory on Machine A.  
You have the option of using the script to create the Web server definition in the configuration of the deployment manager or using the administrative console of the deployment manager to create the Web server definition.
13. Run the `configureWeb_server_name` script on Machine A to create a Web server definition or use the administrative console of the deployment manager to create the definition. You can then use the administrative console to manage the Web server.
14. Propagate the `plugin-cfg.xml` file from the deployment manager on Machine A to the Web server on Machine B using the administrative console. Click **Servers > Web server > Propagate Plug-in**. (Web servers other than IBM HTTP Server require manual propagation.)

You can review common installation scenarios to find a possible match for the topology that you intend to install. Each product installation diagram provides a high-level procedure for installing the components that comprise the topology.

After determining a possible topology, follow the steps in the overall procedure. Useful links to the installation procedures for each installable component are in the list of related topics.

---

## Planning to install Web server plug-ins

This topic describes common installation scenarios and links to component installation procedures for each scenario.

The primary production configuration is an application server on one machine and a Web server on a separate machine. This configuration is referred to as a *remote* configuration. Contrast the remote configuration to the local configuration, where the application server and the Web server are on the same machine.

The Plug-ins installation wizard has four main tasks:

- Installs the binary plug-in module on the Web server machine.
- Configures the Web server configuration file on the Web server machine to point to the binary plug-in module and to the XML configuration file for the binary module.
- Installs a temporary XML configuration file for the binary module (`plugin-cfg.xml`) on the Web server machine in remote scenarios.
- Creates the configuration for a Web server definition on the application server machine. The wizard processes the creation of the Web server definition differently depending on the scenario:

### **Web server plug-in installation for stand-alone application server environments**

- Recommended remote stand-alone Application Server installation:

Creates a configuration script that you run on the application server machine. Install the Web server and its plug-in on a different machine than the application server. This configuration is recommended for a production environment.

- Local stand-alone Application Server installation:

Detects the default profile on a local application server machine and creates the Web server definition for it directly. Install the Web server and its plug-in on the same machine with the application server. This configuration is for development and test environments.

### **Web server plug-in installation for distributed environments (cells)**

- Recommended remote distributed installation:

Creates a configuration script that you run on the application server machine. Install the Web server and its plug-in on a different machine than the deployment manager or managed node. This configuration is recommended for a production environment.

- Local distributed installation:

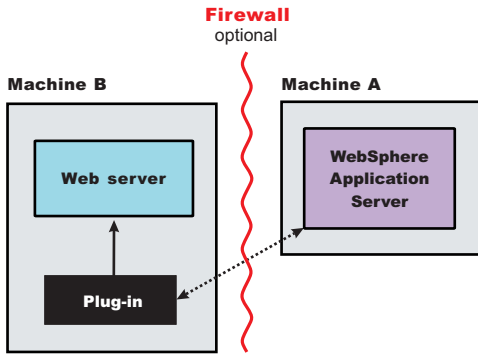
Creates a configuration script that you run when the deployment manager is running. Install the Web server and its plug-in on the same machine with the deployment manager or a managed node. This configuration is for development and test environments.

Select a link to go to the appropriate steps in the following procedure.

- **Set up a remote Web server installation.**


The remote Web server configuration is recommended for production environments.

The remote installation installs the Web server plug-in on the Web server machine when the application server is on a separate machine, such as shown in the following graphic:



### Remote installation scenario

Table 1. Installation and configuration

Step	Machine	Task
1	A	Install your WebSphere Application Server product. See the <i>Installing your application serving environment</i> PDF.
2	A	Create an application server profile. See “Using the Profile creation wizard” on page 45.
3	B	Install IBM HTTP Server or another supported Web server. See your Web server documentation.
4	B	Install the binary plug-in module using the Plug-ins installation wizard. See the <i>Installing your application serving environment</i> PDF.  The script for creating and configuring the Web server is created under the <code>plug-ins_install_root/bin</code> directory.
5	B	Copy the <code>configureWeb_server_name</code> script to Machine A. If one machine is running under Linux or UNIX and the other machine is running under Windows, copy the script from the <code>plug-ins_install_root/bin/crossPlatformScripts</code> directory.
6	A	Paste the <code>configureWeb_server_name</code> script from Machine B to the <code>was_install_root/bin</code> directory on Machine A.
7	A	Run the script from a command line.
8	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
9	B	 <b>UNIX</b> Run the <code>plug-ins_install_root/setupPluginCfg.sh</code> script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
10	B	Run the snoop servlet.  To verify with your own application, regenerate and propagate the <code>plugin-cfg.xml</code> file after installing the application.

### Regeneration of the plugin-cfg.xml file

During the installation of the plug-ins, the temporary `plugin-cfg.xml` file is installed on Machine B in the `plug-ins_install_root/config/web_server_name` directory.

The Web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

To use the real `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next section.

### Propagation of the plugin-cfg.xml file

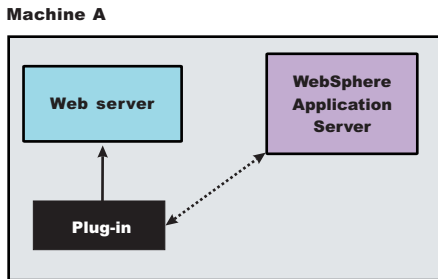
The Web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server 6.0 only.

For all other Web servers, propagate the plug-in configuration file manually. Copy the `plugin-cfg.xml` file from the `profiles_install_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name` directory on Machine A. Paste the file into the `plug-ins_install_root/config/web_server_name` directory on Machine B.

- **Set up a local Web server configuration.**

The local Web server configuration is recommended for a development or test environment.

A local installation includes the Web server plug-in, the Web server, and the application server on the same machine:



### Local installation scenario

Table 2. Installation and configuration

Step	Machine	Task
1	A	Install your WebSphere Application Server product. See the <i>Installing your application serving environment</i> PDF.
2	A	Create an application server profile. See “Using the Profile creation wizard” on page 45.
3	A	Install IBM HTTP Server or another supported Web server. See your Web server documentation.
4	A	Install the binary plug-in module using the Plug-ins installation wizard. See the <i>Installing your application serving environment</i> PDF.  The Web server definition is automatically created and configured during the installation of the plug-ins.
5	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
6	B	<b>UNIX</b> Run the <code>plug-ins_install_root/setupPluginCfg.sh</code> script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
7	B	Run the snoop servlet.  To verify with your own application, regenerate and propagate the <code>plugin-cfg.xml</code> file after installing the application.

### Regeneration of the `plugin-cfg.xml` file

The Web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

The `plugin-cfg.xml` file is generated in the `profiles_install_root/profile_name/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name` directory. The generation occurs when the Web server definition is created.

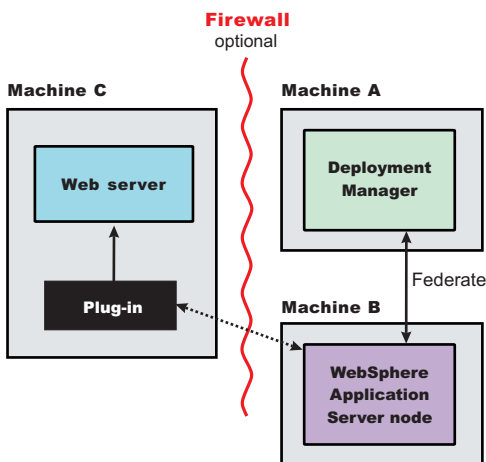
### Propagation of the `plugin-cfg.xml` file

The local file does not require propagation.

- **Set up a remote Web server installation in a cell.**

The remote Web server configuration is recommended for production environments.

The remote installation installs the Web server plug-in on the Web server machine when the application server is on a separate machine, such as shown in the following graphic:




### Remote installation scenario

Table 3. Installation and configuration

Step	Machine	Task
1	A	Install WebSphere Application Server Network Deployment. See the <i>Installing your application serving environment</i> PDF.
2	A	Create a deployment manager profile. See “Using the Profile creation wizard” on page 45.
3	A	Start the deployment manager with the <code>./profiles_install_root/profile_name/bin/startManager.sh</code> command or its Windows equivalent.
4	B	Install WebSphere Application Server Network Deployment. See the <i>Installing your application serving environment</i> PDF.
5	B	Create an application server profile. See “Using the Profile creation wizard” on page 45.
6	B	Federate the node with the <code>./profiles_install_root/profile_name/bin/addNode.sh dmgrhost 8879 -includeapps</code> command or its Windows equivalent.
7	C	Install IBM HTTP Server or another supported Web server. See your Web server documentation.
8	C	Install the binary plug-in module using the Plug-ins installation wizard. See the <i>Installing your application serving environment</i> PDF.  The script for creating and configuring the Web server is created under the <code>plug-ins_install_root/bin</code> directory.
9	C	Copy the <code>configureWeb_server_name</code> script to Machine A.  If one machine is running under Linux or UNIX and the other machine is running under Windows, copy the script from the <code>plug-ins_install_root/bin/crossPlatformScripts</code> directory.
10	A	Paste the <code>configureWeb_server_name</code> script from Machine C to the <code>was_install_root/bin</code> directory on Machine A.

Table 3. Installation and configuration (continued)

Step	Machine	Task
11	A	Run the script from a command line after verifying that the deployment manager is running.  If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the <b>wsadmin</b> command.
12	A/B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.
13	C	 Run the <code>plug-ins_install_root/setupPluginCfg.sh</code> script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
14	C	Run the snoop servlet.  To verify with your own application, regenerate and propagate the <code>plugin-cfg.xml</code> file after installing the application.

### Regeneration of the `plugin-cfg.xml` file

During the installation of the plug-ins, the temporary `plugin-cfg.xml` file is installed on Machine C in the `plug-ins_install_root/config/web_server_name` directory.

The Web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

To use the real `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next section.

### Propagation of the `plugin-cfg.xml` file

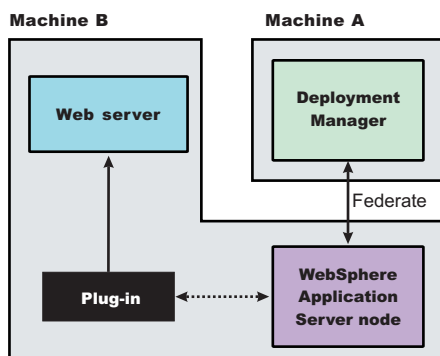
The Web server plug-in configuration service propagates the `plugin-cfg.xml` file automatically for IBM HTTP Server 6.0 only.

For all other Web servers, propagate the plug-in configuration file, by manually copying the `plugin-cfg.xml` file from the `profiles_install_root/profile_name/config/cells/cell_name/nodes/node_name/servers/web_server_name` directory on Machine A to the `plug-ins_install_root/config/web_server_name` directory on Machine C.

- **Set up a local distributed Web server configuration.**

The local Web server configuration is recommended for a development or test environment.


A local distributed installation includes the Web server plug-in, the Web server, and the managed application server on the same machine:





## Local distributed installation scenario

Table 4. Installation and configuration

Step	Machine	Task
1	A	Install WebSphere Application Server Network Deployment. See the <i>Installing your application serving environment</i> PDF.
2	A	Create a deployment manager profile. See “Using the Profile creation wizard” on page 45.
3	A	Start the deployment manager with the <code>profiles_install_root/profile_name/bin/startManager.sh</code> command or its Windows equivalent.
4	B	Install WebSphere Application Server Network Deployment. See the <i>Installing your application serving environment</i> PDF.
5	B	Create an application server profile. See “Using the Profile creation wizard” on page 45.
6	B	Federate the node with the <code>./profiles_install_root/profile_name/bin/addNode.sh dmgrhost 8879 -includeapps</code> command or its Windows equivalent.
7	B	Install IBM HTTP Server or another supported Web server. See your Web server documentation.
8	B	Install the binary plug-in module using the Plug-ins installation wizard. See the <i>Installing your application serving environment</i> PDF.  The script for creating and configuring the Web server is created in the <code>plug-ins_install_root/bin</code> directory.
11	B	After verifying that the deployment manager is running on Machine A, run the <code>configureWeb_server_name</code> script from a command line in the <code>plug-ins_install_root/bin</code> directory on Machine B.  If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters.
12	A/B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.
13	B	 Run the <code>plug-ins_install_root/setupPluginCfg.sh</code> script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
14	B	Run the Snoop servlet.

### Regeneration of the plugin-cfg.xml file

The Web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

The `plugin-cfg.xml` file is generated at the location `profiles_install_root/profile_name/config/cells/cell_name/nodes/node_name/servers/web_server_name` directory, when the Web server definition is created.

Regenerate the `plugin-cfg.xml` file in the Web server definition in the application server whenever the configuration changes. The Web server has immediate access to the file whenever it is regenerated.

When the Web server plug-in configuration service (an administration service) is enabled on Machine A, the `plugin-cfg.xml` file is automatically generated for all Web servers.

### Propagation of the plugin-cfg.xml file

Node synchronization is used to propagate the `plugin-cfg.xml` file from Machine A to Machine B.

When the Web server plug-in configuration service (an administration service) is enabled on Machine A, the `plugin-cfg.xml` file is automatically propagated for all Web servers.

### Alternate configuration

This procedure describes installing the plug-ins on two machines. However, you can perform this procedure on a single machine as shown in the following graphic. A local distributed installation also



includes the Web server plug-in, the Web server, the Application Server, and the deployment manager on the same machine:

You can set up a remote or local Web server by installing Application Server, the Web server, and then the Web server plug-ins.

See your Web server documentation for more information about the files involved in configuring a Web server.

See the *Installing your application serving environment* PDF for information about the logic behind the processing scenarios for the Plug-ins installation wizard.

See the *Installing your application serving environment* PDF for information about how the Plug-ins installation wizard configures supported Web servers.

See See the *Installing your application serving environment* PDF for information about other installation scenarios for installing Web server plug-ins.

---

## Planning to install WebSphere Application Server Clients

This topic helps you examine typical topologies and uses for WebSphere Application Server Clients.

This topic is one in a series of topics described in Chapter 4, “Planning the installation (diagrams),” on page 15. Consider all of the planning scenarios that are mentioned in the parent article to determine the best approach to installing your e-business network. This topic describes installing and using the WebSphere Application Server Clients.

In a traditional client server environment, the client requests a service and the server fulfills the request. Multiple clients use a single server. Clients can also access several different servers. This model persists for Java clients except that now these requests use a client run-time environment.

In this model, the client application requires a servlet to communicate with the enterprise bean, and the servlet must reside on the same machine as the WebSphere Application Server.

The Application Client for WebSphere Application Server, Version 6 now consists of the following models:

- ActiveX application client
- Applet client
- J2EE application client
- Pluggable and thin application clients

The following graphic shows a topology for installing the Application Client and using client applications:

The example shows two types of application clients installed in a topology that uses client applications to access applications and data on Machine A:

- The ActiveX application client on Machine B is a Windows only client that uses the Java Native Interface (JNI) architecture to programmatically access the Java virtual machine (JVM) API. The JVM code exists in the same process space as the ActiveX application (Visual Basic, VBScript, or Active Server Pages (ASP) files) and remains attached to the process until that process terminates.
- The J2EE application client on Machine C is a Java application program that accesses enterprise beans, Java Database Connectivity (JDBC) APIs, and Java Message Service message queues. The application program must configure the execution environment of the J2EE application client and use the Java Naming and Directory Interface (JNDI) name space to access resources.

Use the following procedure as a roadmap for installing the Application Client.

1. Install the WebSphere Application Server product from your product CD on Machine A to establish the core product files.
2. Use the Profile creation wizard to create both stand-alone application server profiles.
3. Use the administrative console of each application server to deploy any user applications.
4. Use the administrative console of each application server to create a Web server configuration for the Web server.
5. Use the administrative console of each application server to regenerate each `plugin-cfg.xml` file in the local Web server configuration.
6. Install the IBM HTTP Server from the product CD on Machine A.
7. Use the Plug-ins installation wizard to install the plug-in for IBM HTTP Server on Machine A.  
The wizard automatically configures the HTTP Server to communicate with the first application server.
8. Install the Application Client from your product CD on Machine B.
  - a. Select the Custom install type.
  - b. Select the ActiveX to EJB Bridge feature.
  - c. Select to add the Java run time to the system path.
  - d. Select the Java run time as the default JRE, which adds the Java run time path to the beginning of the system path.
9. Install the Application Client from your product CD on Machine C.
  - a. Select the Custom install type.
  - b. Select the J2EE application client feature.

This topic can help you plan run-time environments for client applications.

See the *Using administrative clients* PDF for information about creating client applications.

---

## Planning to create application server environments

Application server profiles are the run-time environments for application server processes. This topic describes common scenarios for creating application server profiles and provides links to profile creation procedures for each scenario.

Install the core product files for a WebSphere Application Server product before using the Profile creation wizard to create additional application server run-time environments.

This topic describes how to use the Profile creation wizard to install deployment managers, managed nodes, and stand-alone application servers. Each profile for a deployment manager or an application server is a run-time environment, with data files, configuration files, applications, and an administrative console. The profile for a managed node is a special case that is dependent on the deployment manager profile that owns the managed node.

1. Create a deployment manager, as described in “Using the Profile creation wizard to create a deployment manager” on page 49.

The installation procedure gives you the option of creating a deployment manager during installation. However, you can use the Profile creation wizard to create a deployment manager when one was not created during installation. Or, you can create another deployment manager on a machine where a deployment manager already exists.

2. Create a managed node, as described in “Using the Profile creation wizard to create a custom profile” on page 60.

The installation procedure gives you the option of creating a managed node during installation. However, you can use the Profile creation wizard to create a managed node when one was not created during installation. Or, you can use the wizard to create more managed nodes on a machine where a managed node already exists.

The first part of the process is to install the Network Deployment product to create the core product files. Then you can use the Profile creation wizard to create a managed profile.

The next part of the process is to federate the managed profile into the deployment manager cell. This changes the managed profile into a managed node.

A managed node has a nodeagent process but does not have Sample applications or an application server process in contrast to a stand-alone application server, which has a server1 process and applications, but does not have a nodeagent process.

Start the nodeagent process to allow the administrative console of the deployment manager to create server processes on the managed node.

3. Create a stand-alone application server, as described in “Using the Profile creation wizard to create an application server” on page 73.

The installation procedure gives you the option of creating a stand-alone application server during installation. However, you can use the Profile creation wizard to create a stand-alone application server when one was not created during installation. Or, you can use the wizard to create more stand-alone application servers on a machine where an application server already exists.

4. Create a deployment manager and a managed node on the same machine.

The installation procedure gives you the option of creating a deployment manager and managed node on the same machine during installation. However, you can also use the Profile creation wizard to create a deployment manager and a managed node at any time after installation of the core product files. Or, you can use the wizard to create a managed node on a machine where a deployment manager already exists.

Any time that you create two or more application server processes on one machine, verify that the machine is capable of hosting both processes. See the hardware prerequisites on the IBM WebSphere Application Server supported hardware, software, and APIs Web site.

- a. Create a deployment manager, as described in “Using the Profile creation wizard to create a deployment manager” on page 49.
- b. Create a managed node, as described in “Using the Profile creation wizard to create a custom profile” on page 60.

Use the Profile creation wizard after installing the core product files to create:

- A stand-alone application server environment
- A deployment manager environment
- Managed nodes for the deployment manager cell

After installing the core product files and using the Profile creation wizard to create your e-business environment, you are ready to deploy applications to test the environment.

---

## Example: Choosing a topology for better performance

WebSphere Application Server provides various Workload Management (WLM) topologies. Two topologies were compared to show how the type of topology you choose can affect performance.

In this comparison, Topology A contains a Web server and a WebSphere Application Server plug-in in front of a cluster of WebSphere Application Servers. Each cluster member contains a Web container and an Enterprise JavaBeans (EJB) container. Topology B includes a Web server, a plug-in, and a Web container

in front of a cluster of EJB containers. In both topologies, Object Request Broker (ORB) pass by reference is selected and the backend database is on a dedicated machine.

Result: Topology A had 10% to 20% higher throughput than Topology B when running the Java 2 Platform, Enterprise Edition Benchmark sample for WebSphere (Trade).

**Note:** You can download the Benchmark sample for WebSphere (Trade) from the following Web site:

<http://www-306.ibm.com/software/webservers/appserv/was/performance.html>

Topology A has an advantage because the Web container and EJB container are running in a single Java virtual machine (JVM). In Topology B, the ORB pass by reference option is ignored between the Web container cluster member and the EJB container member. In Topology A, the EJB container uses the same thread passed from the Web container. The request does not have to be passed from one thread in one JVM to another thread in another JVM.

In this test environment, Topology A had the advantage. However, many factors related to the application and environment can influence your results.

---

## Queuing network

WebSphere Application Server contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application. These adjustments help the system achieve maximum throughput while maintaining the overall stability of the system. This group of interconnected components is known as a queuing network. These queues or components include the network, Web server, Web container, EJB container, data source, and possibly a connection manager to a custom back-end system. Each of these resources represents a queue of requests waiting to use that resource. Various queue settings include:

- IBM HTTP Server: MaxClients for UNIX and ThreadsPerChild for Windows NT and Windows 2000 systems described in “Web server tuning parameters” on page 123.
- Web container: **Maximum size** described in “Thread pool settings” on page 201, **MaxKeepAliveConnections** and **MaxKeepAliveRequests** described in “HTTP transport custom properties” on page 219.
- **Tuning Object Request Brokers** explained in “Tuning application servers” on page 256.
- Data source **connection pooling** and **statement cache size** are explained in the *Developing and deploying applications* PDF.

Most of the queues that make up the queuing network are closed queues. A closed queue places a limit on the maximum number of requests present in the queue, while an open queue has no limit. A closed queue supports tight management of system resources. For example, the Web container thread pool setting controls the size of the Web container queue. If the average servlet running in a Web container creates 10MB of objects during each request, a value of 100 for thread pools limits the memory consumed by the Web container to 1GB.

In a closed queue, requests can be active or waiting. An active request is doing work or waiting for a response from a downstream queue. For example, an active request in the Web server is doing work, such as retrieving static HTML, or waiting for a request to complete in the Web container. A waiting request is waiting to become active. The request remains in the waiting state until one of the active requests leaves the queue.

All Web servers supported by WebSphere Application Server are closed queues, as are WebSphere Application Server data sources. You can configure Web containers as open or closed queues. In general, it is best to make them closed queues. EJB containers are open queues. If there are no threads available in the pool, a new one is created for the duration of the request.

If enterprise beans are called by servlets, the Web container limits the number of total concurrent requests into an EJB container, because the Web container also has a limit. The Web container limits the number of total concurrent requests only if enterprise beans are called from the servlet thread of execution. Nothing prevents you from creating threads and bombarding the EJB container with requests. Therefore, servlets should not create their own work threads.

## Queuing and clustering

Cloning application servers can be a valuable asset in configuring highly-scalable production environments, especially when the application is experiencing bottlenecks that are preventing full CPU utilization of symmetric multiprocessing (SMP) servers. When adjusting the WebSphere Application Server system queues in clustered configurations, remember that when a server is added to a cluster, the server downstream receives twice the load.

Two servlet engines are located between a Web server and a data source. It is assumed that the Web server, servlet engines and data source, but not the database, are all running on a single SMP server. Given these constraints, the following queue considerations must be made:

- Double the Web server queue settings to ensure ample work is distributed to each Web container.
- Reduce the Web container thread pools to avoid saturating a system resource like CPU or another resource that the servlets are using.
- Reduce the data source to avoid saturating the database server.
- Reduce Java heap parameters for each instance of the application server. For versions of the Java virtual machine (JVM) shipped with WebSphere Application Server, it is crucial that the heap from all JVMs remain in physical memory. For example, if a cluster of four JVMs is running on a system, enough physical memory must be available for all four heaps.

## Queue configuration tips

The following section outlines a methodology for configuring the WebSphere Application Server queues. Moving the database server onto another machine or providing more powerful resources, for example a faster set of CPUs with more memory, can dramatically change the dynamics of your system.

There are four tips for queuing:

- **Minimize the number of requests in WebSphere Application Server queues.**

In general, requests wait in the network in front of the Web server, rather than waiting in WebSphere Application Server. This configuration only supports those requests that are ready for processing to enter the queuing network. Specify that the queues furthest upstream or closest to the client are slightly larger, and queues further downstream or furthest from the client are progressively smaller.

Queues in the queuing network become progressively smaller as work flows downstream. When 200 client requests arrive at the Web server, 125 requests remain queued in the network because the Web server is set to handle 75 concurrent clients. As the 75 requests pass from the Web server to the Web container, 25 requests remain queued in the Web server and the remaining 50 are handled by the Web container. This process progresses through the data source until 25 user requests arrive at the final destination, the database server. Because there is work waiting to enter a component at each point upstream, no component in this system must wait for work to arrive. The bulk of the requests wait in the network, outside of WebSphere Application Server. This type of configuration adds stability, because no component is overloaded.

You can then use the Edge Server to direct waiting users to other servers in a WebSphere Application Server cluster.

- **Draw throughput curves to determine when the system capabilities are maximized.**

You can use a test case that represents the full spirit of the production application by either exercising all meaningful code paths or using the production application. Run a set of experiments to determine when the system capabilities are fully stressed or when it has reached the saturation point. Conduct

these tests after most of the bottlenecks are removed from the application. The goal of these tests is to drive CPUs to near 100% utilization. For maximum concurrency through the system, start the initial baseline experiment with large queues. For example, start the first experiment with a queue size of 100 at each of the servers in the queuing network: Web server, Web container and data source. Begin a series of experiments to plot a throughput curve, increasing the concurrent user load after each experiment. For example, perform experiments with one user, two users, five, 10, 25, 50, 100, 150 and 200 users. After each run, record the throughput requests per second, and response times in seconds per request. The curve resulting from the baseline experiments resembles the following typical throughput curve shown as follows:

The WebSphere Application Server throughput is a function of the number of concurrent requests present in the total system. Section A, the light load zone, shows that the number of concurrent user requests increases, the throughput increases almost linearly with the number of requests. At light loads, concurrent requests face very little congestion within the WebSphere Application Server system queues. At some point, congestion starts to develop and throughput increases at a much lower rate until it reaches a saturation point that represents the maximum throughput value, as determined by some bottleneck in the WebSphere Application Server system. The most manageable type of bottleneck occurs when the WebSphere Application Server machine CPUs become fully utilized because adding CPUs or more powerful CPUs fixes the bottleneck.

In the heavy load zone or Section B, as the concurrent client load increases, throughput remains relatively constant. However, the response time increases proportionally to the user load. That is, if the user load is doubled in the heavy load zone, the response time doubles. At some point, represented by Section C, the buckle zone, one of the system components becomes exhausted. At this point, throughput starts to degrade. For example, the system might enter the buckle zone when the network connections at the Web server exhaust the limits of the network adapter or if the requests exceed operating system limits for file handles.

If the saturation point is reached by driving CPU utilization close to 100%, you can move on to the next step. If the saturation CPU occurs before system utilization reaches 100%, it is likely that another bottleneck is being aggravated by the application. For example, the application might be creating Java objects causing excessive garbage collection bottlenecks in the Java code.

There are two ways to manage application bottlenecks: remove the bottleneck or clone the bottleneck. The best way to manage a bottleneck is to remove it. You can use a Java-based application profiler, such as Rational Application Developer, Performance Trace Data Visualizer (PTDV), Borland's Optimizeit, JProbe or Jinsight to examine overall object utilization.

- **Decrease queue sizes while moving downstream from the client.**

The number of concurrent users at the throughput saturation point represents the maximum concurrency of the application. For example, if the application saturates WebSphere Application Server at 50 users, using 48 users might produce the best combination of throughput and response time. This value is called the Max Application Concurrency value. Max Application Concurrency becomes the preferred value for adjusting the WebSphere Application Server system queues. Remember, it is desirable for most users to wait in the network; therefore, queue sizes should increase when moving downstream farther from the client. For example, given a Max Application Concurrency value of 48, start with system queues at the following values: Web server 75, Web container 50, data source 45. Perform a set of additional experiments adjusting these values slightly higher and lower to find the best settings.

To help determine the number of concurrent users, view the Servlet Engine Thread Pool and Concurrently Active Threads metric in the Tivoli Performance Viewer.

- **Adjust queue settings to correspond to access patterns.**

In many cases, only a fraction of the requests passing through one queue enters the next queue downstream. In a site with many static pages, a number of requests are fulfilled at the Web server and are not passed to the Web container. In this circumstance, the Web server queue can be significantly larger than the Web container queue. In the previous example, the Web server queue was set to 75, rather than closer to the value of Max Application Concurrency. You can make similar adjustments when different components have different execution times.



For example, in an application that spends 90% of its time in a complex servlet and only 10% of its time making a short JDBC query, on average 10% of the servlets are using database connections at any time, so the database connection queue can be significantly smaller than the Web container queue. Conversely, if the majority of servlet execution time is spent making a complex query to a database, consider increasing the queue values at both the Web container and the data source. Always monitor the CPU and memory utilization for both the WebSphere Application Server and the database servers to verify that the CPU or memory are not saturating.





---

## Chapter 5. Configuring the product after installation

This topic summarizes how to configure the application serving environment.

Use the First steps console to configure and test the WebSphere Application Server environment after installation.

This procedure starts the second and final step of Network Deployment installation. This second step creates and configures server processes by creating profiles.

1. At the end of product installation, select the **Launch the Profile creation wizard** check box to create a deployment manager profile.  
This step creates the deployment manager and the cell. Later steps in this procedure create an Application Server profile and optionally, a custom profile.  
See “Using the Profile creation wizard to create a deployment manager” on page 49.
2. Start the First steps console by selecting the check box on the last panel of the wizard.  
The First steps console for the deployment manager profile can start automatically at the end of profile creation. Select the check box on the last panel of the Profile creation wizard.  
The First steps console is an easy way to start using the product. The console provides one-stop access to the administrative console, Samples Gallery, Profile creation wizard, installation verification test, Migration wizard, and other activities.  
See the description of the “firststeps command” on page 40 for more information.
3. Click **Installation verification** on the First steps console.  
The installation verification test starts the deployment manager process named dmgr and runs several tests to verify that the dmgr process can start without error.  
See “Using the installation verification test” on page 100 for more information.
4. Click **Profile creation wizard** on the First steps console to create an Application Server profile.  
See “Using the Profile creation wizard to create an application server” on page 73.
5. Start the First steps console by selecting the check box on the last panel of the Profile creation wizard.  
This First steps console belongs to the Application Server profile that you just created. Each profile has its own First steps console.
6. Click **Installation verification** on the First steps console.  
The installation verification test starts the new Application Server process named server1 and runs several tests to verify that the server1 process can start without error.
7. Federate the Application Server into the deployment manager cell.  
If both server processes are running, use the administrative console of the deployment manager to add the Application Server node into the cell.  
Point your browser at <http://localhost:9060/ibm/console>, for example, to start the administrative console. Or start it from the First steps console of the deployment manager profile.  
Log in and click **System administration > Nodes > Add Node** and follow the wizard to add the node into the cell. You can use localhost for the Host field if both processes are on the same machine. The SOAP port for the Application Server node is 8880 unless you changed the port during profile creation.  
If the deployment manager is running, you can use the **addNode** command instead. See the *Administering applications and their environment* PDF for a description of this command.
8. **Optional:** Click **Profile creation wizard** on the First steps console to create a custom profile.  
Verify that the deployment manager is running. The Profile creation wizard can federate the custom node for you if the deployment manger is running.  
Supply the host name and the SOAP port for the deployment manager while creating the custom profile.

Choose to federate the custom node into the deployment manager cell. A custom profile must be part of a cell.

Use the deployment manager to customize the node at your leisure. Add servers, add clusters, and install applications on the node, for example.

See “Using the Profile creation wizard to create a custom profile” on page 60.

This procedure results in configuring and testing the Application Server environment.

See “Planning to install Network Deployment” on page 16 for diagrams of topologies that you can create using the First steps console and the Profile creation wizard.

---

## firststeps command

The **firststeps** command starts the First steps console.

### The First steps console

The First steps console is a post-installation ease-of-use tool for directing WebSphere Application Server elements from one place. Options display dynamically on the First steps console, depending on features you install. With all of the options present, you can use the First steps console to start or stop the application server, verify the installation, access the information center, access the administrative console, launch the Migration wizard, or access the Samples gallery.

The First steps console for the Network Deployment product has several forms. A First steps console exists for the product itself before the creation of any profiles. This version lets you start the Profile creation wizard to get started defining a deployment manager and application servers for the cell. You can also define stand-alone application servers. Each stand-alone application server has its own First steps console. Each deployment manager has its own First steps console.

A prompt to launch the First steps console displays on the last panel of the Profile creation wizard.

You can also start the First steps console from the command line as described later.

The First steps console for the Network Deployment product has several forms. A console exists for the product itself before the creation of any profiles. Options that display on the First steps console in the core product files of the Network Deployment product include the following entries:

#### Profile creation wizard

This option starts the Profile creation wizard. The wizard lets you create a deployment manager profile, an application server profile, or a custom profile. A *profile* consists of files that define the run-time environment for the deployment manager or the application server. Each environment has its own administrative interface. A custom profile is an exception. A custom profile is an empty node that you can federate into a deployment manager cell and customize. No default server processes or applications are created for a custom profile.

Each deployment manager or application server profile has its own First steps console. The location of the command to launch the First steps console is within the set of files in the profile. A prompt to launch the First steps console that is associated with a profile displays on the last panel of the Profile creation wizard.

#### Information center for WebSphere Application Server

This option links you to the online information center at the <http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp> IBM Web address.

## Migration wizard

This option starts the Migration wizard. The Migration wizard is a new graphical interface to the migration tools. The migration tools are described in the *Migrating, coexisting, and interoperating* PDF.

**Exit** This option closes the First steps console.

In addition to the generic First steps console for the Network Deployment product, other First steps consoles exist for the deployment manager profile and the application server profile that the Network Deployment product can create. Options that display on the First steps console for a deployment manager include the following entries:

## Installation verification

This option starts the installation verification test. The test consists of starting and monitoring the deployment manager during its start up.

If this is the first time that you have used the First steps console since creating a deployment manager profile, click **Installation verification** to verify that all is well with your installation. The verification process starts the deployment manager.

If you select the **Installation verification** option, the **Start the deployment manager** option is grayed out while the IVT is running.

The IVT provides the following useful information about the deployment manager:

- The deployment manager server name: dmgr
- The name of the profile
- The profile file path
- The type of profile: dmgr
- The cell name
- The node name
- The current encoding
- The port number for the administrative console
- Various informational messages that include the location of the SystemOut.log file and how many errors are listed within the file
- A completion message

## Start the deployment manager

This option displays when you use the Profile creation wizard to create a deployment manager. This option toggles to **Stop the deployment manager** when the deployment manager is running.

After selecting the **Start the deployment manager** option, an output screen displays with status messages. The success message informs you that the deployment manager is open for e-business. Then the menu item changes to **Stop the deployment manager**.

If you select the **Start the deployment manager** option, the **Installation verification** option is grayed out while the deployment manager is running.

## Administrative console

This option is grayed out until the application server is running.

The administrative console is a configuration editor that runs in a Web browser. The administrative console lets you work with XML configuration files for the deployment manager and all of the application servers that are in the cell. To launch the administrative console, click **Administrative console**. You can also point your browser to <http://localhost:9060/ibm/console> to start the administrative console. Substitute your own host name in the address if the localhost variable does not resolve correctly. As the administrative console opens, it prompts you for a login name. This is not a security item, but merely a tag to identify configuration changes that you make during the session. Secure signon is also available.

### Profile creation wizard

This option starts the Profile creation wizard. The wizard lets you create a deployment manager profile, an application server profile, or a custom profile. A *profile* consists of files that define the run-time environment for the deployment manager or the application server. Each environment has its own administrative interface. A custom profile is an exception. A custom profile is an empty node that you can federate into a deployment manager cell and customize. No default server processes or applications are created for a custom profile.

Each deployment manager profile has its own First steps console. The location of the command to launch the First steps console is within the set of files in the profile. A prompt to launch the First steps console that is associated with a profile displays on the last panel of the Profile creation wizard.

### Information center for WebSphere Application Server

This option links you to the online information center at the <http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp> IBM Web address.

### Migration wizard

This option starts the Migration wizard. The Migration wizard is a new graphical interface to the migration tools. The migration tools are described in the *Migrating, coexisting, and interoperating* PDF.

**Exit** This option closes the First steps console.

In addition to the generic First steps console and the First steps console for the deployment manager, another First steps console exists for stand-alone application servers that you create with the Profile creation wizard. Options that display on the First steps console for an application server profile include the following entries:

### Installation verification

This option starts the installation verification test. The test consists of starting and monitoring the application server during its start up.

If this is the first time that you have used the First steps console since creating an application server profile, click **Installation verification** to verify that all is well with your installation. The verification process starts the application server.

If you select the **Installation verification** option, the **Start the server** option is grayed out while the IVT is running.

The IVT provides the following useful information about the application server:

- The server name: server1
- The name of the profile
- The profile file path
- The type of profile: default
- The cell name
- The node name
- The current encoding
- The port number for the administrative console
- Various informational messages that include the location of the SystemOut.log file and how many errors are listed within the file
- A completion message

### Start the server

This option toggles to **Stop the server** when the application server is running.

After selecting the **Start the server** option, an output screen displays with status messages. The success message informs you that the server is open for e-business. Then the menu item changes to **Stop the server**.

If you select the **Start the server** option, the **Installation verification** option is grayed out while the application server is running.

### Administrative console

This option is grayed out until the application server is running.

The administrative console is a configuration editor that runs in a Web browser. The administrative console lets you work with XML configuration files for the deployment manager and all of the application servers that are in the cell. To launch the administrative console, click **Administrative console**. You can also point your browser to `http://localhost:9060/ibm/console` to start the administrative console. Substitute your own host name in the address if the localhost variable does not resolve correctly. As the administrative console opens, it prompts you for a login name. This is not a security item, but merely a tag to identify configuration changes that you make during the session. Secure signon is also available.

### Profile creation wizard

This option starts the Profile creation wizard. The wizard lets you create a deployment manager profile, an application server profile, or a custom profile. A *profile* consists of files that define the run-time environment for the deployment manager or the application server. Each environment has its own administrative interface. A custom profile is an exception. A custom profile is an empty node that you can federate into a deployment manager cell and customize. No default server processes or applications are created for a custom profile.

Each application server profile has its own First steps console. The location of the command to launch the First steps console is within the set of files in the profile. A prompt to launch the First steps console that is associated with a profile displays on the last panel of the Profile creation wizard.

### Samples gallery

This option starts the Samples gallery. The option is grayed out until you start the application server. The option displays when you have installed the Samples during installation.

From the First steps console, click **Samples gallery** to explore the application Samples. Alternatively you can point your browser directly to `http://localhost:9080/WSsamples`. Substitute your own host name in the address if the localhost variable does not resolve correctly. The Web address is case sensitive. Substitute your own host name in the address.

### Information center for WebSphere Application Server

This option links you to the online information center at the `http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp` IBM Web address.

### Migration wizard



This option starts the Migration wizard. The Migration wizard is a new graphical interface to the migration tools. The migration tools are described in the *Migrating, coexisting, and interoperating* PDF.

**Exit** This option closes the First steps console.

The First steps console for a managed node has the same options as the generic First steps console for the Network Deployment product.

### Location of the command file

The location of the `firststeps.sh` or `firststeps.bat` script for any profile is:

-  `install_root/profiles/profile_name/firststeps/firststeps.sh`
-  `install_root\profiles\profile_name\firststeps\firststeps.bat`

The location of the **firststeps** command for the generic First steps console for the Network Deployment product is:

- **UNIX** `install_root/firststeps/firststeps.sh`
- **Windows** `install_root\firststeps\firststeps.bat`

## Parameters

No parameters are associated with this command.

## Syntax for the firststeps command





Use the following syntax for the command:

- **UNIX** `./firststeps.sh`
- **Windows** `firststeps.bat`

## Usage tips

The following links exist on one of the First steps consoles for the Network Deployment product. Not all links display on each First steps console. For example, the First steps console for the deployment manager does not have the Samples Gallery option or the Start the server option.

Option	Link
<b>Installation verification</b>	<p>Calls the <b>ivt</b> command.</p> <p>The location of the installation verification test varies per platform:</p> <ul style="list-style-type: none"> <li>• <b>UNIX</b> <code>install_root/profiles/profile_name/bin/ivt.sh</code></li> <li>• <b>Windows</b> <code>install_root\profiles\profile_name\bin\ivt.bat</code></li> </ul>
<b>Start the server</b>	<p>Calls the <b>startServer</b> command.</p> <p>The location of the <b>startServer</b> command varies per platform:</p> <ul style="list-style-type: none"> <li>• <b>UNIX</b> <code>install_root/profiles/profile_name/bin/startServer.sh server1</code></li> <li>• <b>Windows</b> <code>install_root\profiles\profile_name\bin\startServer.bat server1</code></li> </ul> <p>When you have more than one application server on the same machine, the command starts the same application server that is associated with the First steps console.</p>
<b>Stop the server</b>	<p>Calls the <b>stopServer</b> command.</p> <p>The location of the <b>stopServer</b> command varies per platform:</p> <ul style="list-style-type: none"> <li>• <b>UNIX</b> <code>install_root/profiles/profile_name/bin/stopServer.sh server1</code></li> <li>• <b>Windows</b> <code>install_root\profiles\profile_name\bin\stopServer.bat server1</code></li> </ul>
<b>Start the deployment manager</b>	<p>Calls the <b>startManager</b> command.</p> <p>The location of the <b>startManager</b> command varies per platform:</p> <ul style="list-style-type: none"> <li>• <b>UNIX</b> <code>install_root/profiles/profile_name/bin/startManager.sh</code></li> <li>• <b>Windows</b> <code>install_root\profiles\profile_name\bin\startManager.bat</code></li> </ul> <p>When you have more than one deployment manager on the same machine, the command starts the same deployment manager that is associated with the First steps console.</p>
<b>Stop the deployment manager</b>	<p>Calls the <b>stopManager</b> command.</p> <p>The location of the <b>stopManager</b> command varies per platform:</p> <ul style="list-style-type: none"> <li>• <b>UNIX</b> <code>install_root/profiles/profile_name/bin/stopManager.sh</code></li> <li>• <b>Windows</b> <code>install_root\profiles\profile_name\bin\stopManager.bat</code></li> </ul>

Option	Link
<b>Administrative console</b>	<p>Opens the default browser to the <a href="http://localhost:9060/ibm/console">http://localhost:9060/ibm/console</a> Web address.</p> <p>When you have more than one application server on the same machine, the port varies. The First steps console starts the administrative console that is associated with the First steps console.</p>
<b>Profile creation wizard</b>	<p>Calls the <b>pctplatform</b> command.</p> <p>The command is in the <code>install_root/bin/ProfileCreator</code> directory. The name of the command varies per platform:</p> <ul style="list-style-type: none"> <li>• <code>pctAIX.bin</code></li> <li>• <code>pctHPUX.bin</code></li> <li>• 64-bit platforms: <code>pctHPUXIA64.bin</code></li> <li>• <code>pctLinux.bin</code></li> <li>• 64-bit platforms: <code>pctLinuxIA64.bin</code> S/390 platforms: <code>pctLinux390.bin</code></li> <li>• Power platforms: <code>pctLinuxPPC.bin</code></li> <li>• <code>pctSolaris.bin</code></li> <li>•  <code>pctWindows.exe</code></li> <li>•  64-bit platforms: <code>pctWindowsIA64.exe</code></li> </ul>
<b>Samples Gallery</b>	<p>Opens the default browser to the <a href="http://localhost:9080/WSsamples">http://localhost:9080/WSsamples</a> Web address.</p> <p>If you do not install Samples, the option does not appear on the First steps console. If you do not install the Samples during the initial installation of the product, the option does not display on the First steps console. You can perform an incremental installation to add the Samples feature. After adding the Samples, the options displays on the First steps console.</p> <p>When you have more than one profile on the same machine, the port varies. The First steps console starts the Samples gallery that is associated with the First steps console.</p>
<b>Information center for WebSphere Application Server products</b>	<p>Opens the default browser to the online information center at the <a href="http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp">http://publib.boulder.ibm.com/infocenter/ws60help/index.jsp</a> Web address.</p>
<b>Migration wizard</b>	<p>Calls the <b>migration</b> command.</p> <p>The location of the <b>migration</b> command is:</p> <ul style="list-style-type: none"> <li>•  <code>install_root/bin/migration.sh</code></li> <li>•  <code>install_root\bin\migration.bat</code></li> </ul> <p>The migration tools are also in the <code>/migration</code> folder on the product disc.</p>

## Using the Profile creation wizard

This topic describes how to create run-time environments for WebSphere Application Server. Each run-time environment is created within a *profile*. A profile is the set of files that define the run-time environment. The Profile creation wizard creates the profile for each run-time environment.

Before using the Profile creation wizard, install the core product files.

The Profile creation wizard is the wizard interface to the profile creation tool, `wasprofile`. See the description of the “`wasprofile` command” on page 86 for more information.



An error can occur when you have not provided enough system temporary space to create a profile. Verify that you have a minimum of 40 MB of temp space available before creating a profile.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

**Important:**

You must have 30 MB of available disk space in the directory where you create a deployment manager profile.

You must have 10 MB of available disk space in the directory where you create a custom profile.

Manually verify that the required space for creating a profile is available on AIX. A known problem in the underlying InstallShield for Multiplatforms (ISMP) code prevents proper space checking on AIX systems at the time that the product disc was created.

**Important:** Concurrent profile creation is not supported at this time for one set of core product files. Concurrent attempts to create profiles result in a warning about a profile creation already in progress.

The installation procedure for the Network Deployment product does not create a run-time environment by default because three possibilities exist. After installing the core product files for the Network Deployment product, use the Profile creation wizard to create any combination of the following three profiles to have an operational run-time environment:

- A deployment manager profile.

See “Using the Profile creation wizard to create a deployment manager” on page 49. Create this run-time environment first, to create the administrative node for a multinode, multimachine group of application server nodes that you create later. This logical group of application server processes is known as a *cell*.

- An application server profile.

See “Using the Profile creation wizard to create an application server” on page 73.

When you create the application server profile, a default server1 process is created. The process has all of the Sample applications and its own administrative console. You can federate the server1 node into the deployment manager cell with the **addNode** command or from the administrative console of the deployment manager. The server1 process must be running to begin the federation from the deployment manager.

If you include all of the applications from the application server, the act of federation installs the applications on the deployment manager where they can be redeployed.

Optionally, you can create stand-alone application servers by creating an application server profile and not federating the node. If you remove a federated application server node from a deployment manager, the application server returns to its original configuration, which is a stand-alone application server.

- A custom profile.

See “Using the Profile creation wizard to create a custom profile” on page 60. A custom profile is an empty node that you can customize through the deployment manager to include application servers, clusters, or other Java processes, such as a messaging server. Create a custom profile on a distributed machine and add the node into the deployment manager cell to get started customizing the node.

Each use of the Profile creation wizard or the **wasprofile** command line tool creates one profile.



1. Install the product to create the core product files.
2. Start the Profile creation wizard to create a new run-time environment.



Several ways exist to start the wizard. The initial way to start the wizard is at the end of installation by selecting the check box to launch the Profile creation wizard.

One way to start the wizard is to issue the command directly from a command line.

The command is in the *install\_root/bin/ProfileCreator* directory. The name of the command varies per platform:




- `pctAIX.bin`
- `pctHPUX.bin`
- 64-bit platforms: `pctHPUXIA64.bin`
- `pctLinux.bin`
- 64-bit platforms: `pctLinux390.bin` S/390 platforms: `pctLinux390.bin`
- Power platforms: `pctLinuxPPC.bin`
- `pctSolaris.bin`
-  `pctWindows.exe`
-  64-bit platforms: `pctWindowsIA64.exe`

Another way to start the Profile creation wizard is to select the wizard from the First steps console.

a. Open a command window.

b. Change directories to the `firststeps` directory in the installation root directory:

The installation root varies by platform:

- `./usr/IBM/WebSphere/AppServer/firststeps`
  - `.../opt/IBM/WebSphere/AppServer/firststeps`
  -  `C:\Program Files\IBM\WebSphere\AppServer\firststeps`
- c. Issue the **firststeps** command to start the console:
-  `./firststeps.sh`
  -  `firststeps.bat`
- d. Select the Profile creation wizard option on the console.

The Profile creation wizard is an InstallShield for Multiplatforms application. The wizard loads the Java 2 SDK and then displays its Welcome panel.

See the description of the “firststeps command” on page 40 for more information.

3. Create a profile.

You can create profiles in any order. However, to create a functioning cell in the shortest possible time, create a deployment manager profile first. Then create an application server profile and add it to the deployment manager cell. You now have a functioning cell with a managed node that you can manage from the administrative console of the deployment manager.

A custom profile requires a greater amount of customization. When you create a custom profile, you must use the **addNode** command to federate it into the deployment manager cell. In contrast to an application server profile, a custom profile does not have a default application server on its node. The `server1` application server does not exist by default on the custom node. Nor are there any default applications on the custom node. Use the administrative console of the deployment manager to customize the empty node for production or other uses. You can create application servers or clusters on the node, for example.

Create any of the following profiles:

- Create a deployment manager.

See “Using the Profile creation wizard to create a deployment manager” on page 49.

Create a deployment manager to establish a cell. Although you can create an application server profile and use it as a stand-alone application server, you must have a deployment manager to use a custom profile. So there is no point in creating a custom profile until you have created a deployment manager.

- Create an application server profile, as described in “Using the Profile creation wizard to create an application server” on page 73.

Federate the application server into the deployment manager cell to create a federated server1 application server. A stand-alone application server has default applications. You can include the applications as you federate the application server to install the default applications on the deployment manager.

Two methods exist for federating application servers into a deployment manager cell:

- Start the deployment manager and the application server and use the administrative console of the deployment manager to federate the node. Click **System administration > Nodes > Add node > Managed node > Next** and identify the host name and the SOAP port of the machine where you created the application server.
- Start the deployment manager. Go to the *install\_root/profiles/profile\_name/bin* directory of the application server and issue the **addNode** command. See the *Administering applications and their environment* PDF for more information.

- Create a custom profile, as described in “Using the Profile creation wizard to create a custom profile” on page 60.

The first part of the process is to install the Network Deployment product to create the core product files. Then you can use the Profile creation wizard to create a managed profile.

The next part of the process is to federate the managed profile into the deployment manager cell. This changes the custom profile into a managed node.

After federation, a custom profile has a nodeagent process but does not have an application server process. Contrast this situation to an application server profile that has a server1 process, but does not have a nodeagent process until you federate the node.

Start the nodeagent process to allow the administrative console of the deployment manager to create server processes on the managed node.

Two methods exist for federating a custom node into a deployment manager cell:

- Federate the custom node during custom profile creation, either with the wizard or as the wizard runs in silent mode.

The deployment manager must be running and accessible at the host address you supply. The deployment manager must also use the default JMX connector type, which is SOAP. The deployment manager must not have security enabled. If any of these conditions are not met, do not federate the custom profile as you create it, but federate it later with the **addNode** command. Otherwise, you create a faulty profile that you must move or delete from the profiles repository directory before creating another profile.

- Use the **addNode** command to federate the custom node after you create the custom profile.
  - a. Start the deployment manager.
  - b. Go to the *install\_root/profiles/profile\_name/bin* directory of the custom profile and issue the **addNode** command.
  - c. Within the same directory, issue the **startNode** command. See the *Administering applications and their environment* PDF for more information.

After federation, go to the administrative console of the deployment manager to customize the empty node.

- Create a deployment manager and a managed node on the same machine:
  - a. Create a deployment manager, as described in “Using the Profile creation wizard to create a deployment manager” on page 49.
  - b. Start the deployment manager with the **startManager** command. See the *Administering applications and their environment* PDF for more information.
  - c. Create an application server profile, as described in “Using the Profile creation wizard to create an application server” on page 73.

- d. Start the application server with the **startServer** command. See the *Administering applications and their environment* PDF for more information.
- e. Use the administrative console of the deployment manager to add the application server node into the deployment manager cell. Click **System administration > Nodes > Add node > Managed node > Next** and identify the host name of the machine and the SOAP port of the application server.

The SOAP port is identified in the `install_root/profiles/profile_name/config/cells/cell_name/nodes/node_name/serverindex.xml` file. You can also use the administrative console of the application server to view its SOAP port setting. Click **Servers > Application servers > server1 > Ports**. The SOAP port is usually the second port in the list.

Select the check box to include applications that are installed on the application server. The default application has the snoop servlet and the hitcount servlet, which are useful for testing. Adding the stand-alone application server node to the deployment manager node changes the server1 process into a managed node. Use the administrative console of the deployment manager to configure the server1 process.

You can also create a custom profile and federate the node during profile creation, or use the **addNode** command to federate the empty node into the cell after the custom profile exists. A managed node created from a custom profile requires customization to create an application server and install applications. Use the administrative console of the deployment manager to configure the custom node.

See the *Administering applications and their environment* PDF to learn more about the command-line alternative method of creating a profile, and to see examples of using the command.

See Chapter 4, “Planning the installation (diagrams),” on page 15 for examples of configurations that you can create by creating profiles.

## Using the Profile creation wizard to create a deployment manager

This topic describes creating a run-time environment for a deployment manager.

Before using the Profile creation wizard, install the core product files.

The Profile creation wizard is the wizard interface to the profile creation tool, `wasprofile`. See the description of the “`wasprofile` command” on page 86 for more information.

An error can occur when you have not provided enough system temporary space to create a profile. Verify that you have a minimum of 40 MB of temp space available before creating a profile.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

You must have 30 MB of available disk space in the directory where you create a deployment manager profile.

You must have 10 MB of available disk space in the directory where you create a custom profile.

Manually verify that the required space for creating a profile is available on AIX. A known problem in the underlying InstallShield for Multiplatforms (ISMP) code prevents proper space checking on AIX systems at the time that the product disc was created.

After installing the core product files, you must create one of the following profiles to have an operational run-time environment:

- An application server that is described in this topic.

An application server profile has a default server (which is server1), the default application that includes the snoop servlet and the hitcount servlet, and application Samples. You can federate the application server or use it as a stand-alone application server.

- A deployment manager.

See “Using the Profile creation wizard to create a deployment manager” on page 49. The deployment manager provides a single administrative interface to a logical group of application servers on one or more machines.

- A custom profile that you must federate to make operational.

See “Using the Profile creation wizard to create a custom profile” on page 60. A custom profile is an empty node that you can customize to include application servers, clusters, or other Java processes, such as a messaging server.

This procedure describes creating a deployment manager profile. The procedure uses the graphical user interface provided by the Profile creation wizard. You can use the Profile creation wizard in silent mode with a response file, without the graphical user interface. See “responsefile.pct.NDdmgrProfile.txt” on page 53 for examples of using the Profile creation wizard in silent mode.



You can also use the **wasprofile** command to create a deployment manager. See the description of the “wasprofile command” on page 86 for more information.

1. Start the Profile creation wizard to create a new run-time environment.

Several ways exist to start the wizard. The initial way to start the wizard is at the end of installation by selecting the check box to launch the Profile creation wizard.

One way to start the wizard is to issue the command directly from a command line.

The command is in the *install\_root/bin/ProfileCreator* directory. The name of the command varies per platform:


- `./pctAIX.bin`
- `./pctHPUX.bin`
- 64-bit platforms: `pctHPUXIA64.bin`
- `./pctLinux.bin`
- 64-bit platforms: `pct.bin` S/390 platforms: `pctLinux390.bin`
- Power platforms: `pctLinuxPPC.bin`
- `./pctSolaris.bin`
-  `pctWindows.exe`
-  64-bit platforms: `pctWindowsIA64.exe`

Another way to start the Profile creation wizard is to select the wizard from the First steps console.



- a. Open a command window.

- b. Change directories to the `firststeps` directory in the installation root directory:

The installation root varies by platform:

- `./usr/IBM/WebSphere/AppServer/firststeps`
- `.../opt/IBM/WebSphere/AppServer/firststeps`
-  `C:\Program Files\IBM\WebSphere\AppServer\firststeps`

- c. Issue the **firststeps** command to start the console:

-  `./firststeps.sh`
-  `firststeps.bat`

- d. Select the Profile creation wizard option on the console.

The Profile creation wizard is an InstallShield for Multiplatforms application. The wizard loads the Java 2 SDK and then displays its Welcome panel.

See the description of the “firststeps command” on page 40 for more information.

2. Click **Next** on the Welcome panel.  
The wizard displays the Profile type selection panel.
3. Select the type of profile to create and click **Next**. In this example, select the option for creating a deployment manager and click **Next**.  
The wizard displays the Profile name panel.
4. Specify a name for the profile, then click **Next**.  
**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as \*&?
- Slashes (/) or (\)

Double-byte characters are allowed.

### The default profile

The first profile that you create on a machine is the default profile. The default profile is the default target for commands issued from the `bin` directory in the product installation root. When only one profile exists on a machine, every command works on the only server process in the configuration.

### Addressing a profile in a multi-profile environment

When two or more profiles exist on a machine, certain commands require that you specify the profile to which the command applies. These commands use the `-profileName` parameter to identify which profile to address. You might find it easier to use the commands that in the `bin` directory of each profile.

A command in the `profiles/profile_name/bin` directory has two lines. The first line sets the `WAS_USER_SCRIPT` environment variable for the command window. The variable sets up the command environment to address the profile. The second line calls the actual command in the `install_root/bin` directory.

The actual command queries the command shell to determine the calling profile and to autonomically address the command to the calling profile.

The wizard then displays the Profile directory panel.

5. Accept the default directory, specify a non-default location, or click **Browse** to select a different location. Click **Next**.

If you click **Back** and change the name of the profile, you must manually change the name on this panel when it displays again.

The wizard displays the Node, host, and cell name panel.

6. Specify a unique node name, the actual host name of the machine, and a unique cell name. Click **Next**.

The deployment manager node has the following characteristics.


Field name	Default value	Constraints	Description
Node name	The name of your machine, or a unique derivation of the machine name.	Use a unique name for the deployment manager.  See the following note.	The name is used for administration within the deployment manager cell.
Host name	The DNS name of your machine.	The host name must be addressable through your network.  See the following note.	Use the actual DNS name or IP address of your machine to enable communication with your machine. See additional information about the host name that follows this table.

Field name	Default value	Constraints	Description
Cell name	The arbitrary name of the deployment manager cell. The cell is a logical grouping of managed nodes, under the control of the deployment manager.	Use a unique name for the deployment manager cell. If you plan to migrate a V5 deployment manager cell to this V6 deployment manager, use the same cell name as the V5 deployment manager.  See the following note.	All federated nodes become members of the deployment manager cell, which you name in this panel.

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

#### Node and cell name considerations

 The profiles directory path must be no longer than 80 characters.

#### Host name considerations

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the hostName property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as xmachine.manhattan.ibm.com
- The default short DNS host name string, such as xmachine
- Numeric IP address, such as 127.1.255.3

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected from the network. Define the short name to 127.0.0.1 (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.

A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A



format disadvantage is that the numeric IP address is fixed. You must change the setting of the `hostName` property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

After specifying deployment manager characteristics, the wizard displays the Port value assignment panel.

7. Verify that the ports specified for the deployment manager are unique and click **Next**.

**Windows** The wizard displays the Windows service definition panel.

8. **Windows** Choose whether to run the `dmgr` process as a Windows service on a Windows platform and click **Next**.

Version 6 attempts to start Windows services for `dmgr` processes started by a **startManager** command. For example, if you configure a deployment manager as a Windows service and issue the **startManager** command, the **wasservice** command attempts to start the defined service.

If you chose to install a local system service, you do not have to specify your user ID or password. If you create a specified user type of service, you must specify the user ID and the password for the user who is to run the service. The user must have *Log on as a service* authority for the service to run properly.

To perform this installation task, the user ID must not have spaces in its name. The ID must also belong to the administrator group and must have the advanced user rights *Act as part of the operating system* and *Log on as a service*. The Installation wizard grants the user ID the advanced user rights if it does not already have them, if the user ID belongs to the administrator group.

You can also create other Windows services after the installation is complete, to start other server processes. See “Automatically restarting server processes” on page 235 for more information.

The wizard displays the Profile summary panel.

9. Click **Next** to create the deployment manager or click **Back** to change the characteristics of the deployment manager.

The wizard displays a Status panel during the creation of the profile. When the installation is complete, the wizard displays the Profile creation is complete panel.

10. Click **Finish** to exit and then click **Profile creation wizard** on the First steps console to start the wizard again, if you intend to create an application server profile.

You can create a deployment manager profile. The node within the profile has a deployment manager named `dmgr`.

Refer to the description of the “`wasprofile` command” on page 86 to learn about creating this type of profile using a command instead of a wizard.

Create an application server profile and add the node into the cell. Then you are ready to deploy an application.

Deploy an application to get started!

### **responsefile.pct.NDdmgrProfile.txt**

This topic describes the response file for creating a deployment manager profile.

Create a deployment manager profile with an options response file after logging on as root on a Linux or UNIX platform, or as a user that belongs to the administrator group on a Windows platform. Some steps of the installation procedure on a Windows platform require the user to belong to the administrator group and to have the advanced user rights *Act as part of the operating system* and *Log on as a service*.

The response file is shipped with default values.

A common use for an options file is to run the Installation wizard in silent mode, which is referred to as installing silently. The wizard reads the options file to determine responses and does not display the graphical user interface. Issue the following command to use a copy of the options file named `myresponsefile.txt` for a silent installation:

```
pctWindows.exe -options "myresponsefile.txt" -silent
```

### Avoiding the use of the `-silent` option within the options response file

A problem occurs when the `-silent` option exists in the file. The file works with the option during a direct call to the profile creation wizard, but fails when called from a silent product installation. See the *Installing your application serving environment* PDF for information about creating a profile silently during a silent product installation.

The option is unnecessary. Avoid using the option to avoid problems.

### Response file locations

The example options response files are in two locations.

#### Example files:

- `responsefile.pct.NDdmgrProfile.txt`
- `responsefile.pct.NDmanagedProfile.txt`
- `responsefile.pct.NDstandAloneProfile.txt`

#### Location:

Table 5. Option response file locations

Product disc location	Installed location
/WAS directory	<code>install_root/bin/ProfileCreator</code> directory

Use the file on the product disc to install the Network Deployment product silently and create a profile.

After installing the Network Deployment product, you can use the installed response file with the `-options` parameter on the Profile creation wizard command.

### Required disk space

Profile	Required disk space	Required temp space
Deployment manager profile	30 MB	40 MB
Custom profile	10 MB	40 MB
Application server profile	200 MB	40 MB

### Creating an operational environment during product installation

Version 6 installation of the Network Deployment product is a two-step process:

1. Installing the core product files and feature files.
2. Creating a deployment manager profile, a custom profile, or an application server profile.

The sample options response file, `responsefile.nd.txt`, controls the first part of the installation and can also start the second part of the installation. To create a profile as part of installing the core product files, use the option in the `responsefile.nd.txt` file that identifies the response file for creating a profile. The profile response file lets you use the Profile creation wizard silently.



To edit and use the appropriate response file for creating a profile, perform the following procedure:

1. Copy the appropriate file from the WAS directory on the product disc to a place that you can easily identify on your machine. The example files are:

To create a:	Copy the following response file:
Deployment manager profile	responsefile.pct.NDdmgrProfile.txt
Custom profile	responsefile.pct.NDmanagedProfile.txt
Application server profile	responsefile.pct.NDstandAloneProfile.txt

2. Edit the file to customize the values for your installation.
3. Verify that no `-silent` option exists in the response file for the Profile creation wizard. If the option exists, the profile is not created.
4. Save the file.
5. Edit the `responsefile.nd.txt` file to identify the location and name of the profile response file. Change the value of the `-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation` option to identify the file. For example:
 

```
-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation=
"/opt/IBM/WebSphere/MyOptionFiles/customProfile.txt"
```
6. Start the installation. For example:
 

```
install -options "myresponsefile.txt" -silent
```
7. After the installation, examine the logs for success.

### Creating a profile after installation

Version 6 installation of the Network Deployment product is a two-step process:

1. Installing the core product files and feature files
2. Creating a deployment manager profile, a custom profile, or an application server profile

When the core product files exist, create a profile at any time using the Profile creation wizard. Start the wizard from the First steps console or directly using the Profile creation wizard command.

You can also use one of the following sample options response files for profiles to create a profile silently using the Profile creation wizard in silent mode. To edit and use the appropriate response file for creating a profile, perform the following procedure:

1. Copy the appropriate file from the `install_root/bin/ProfileCreator` directory to a place that you can easily identify on your machine. The example files are:



To create a profile for a:	Copy the following response file:
Deployment manager	responsefile.pct.NDdmgrProfile.txt
Managed node	responsefile.pct.NDmanagedProfile.txt
Stand-alone application server	responsefile.pct.NDstandAloneProfile.txt

For example, copy the file as `my_options_file.txt`

2. Edit the file to customize the values for your installation.
3. Save the file.
4. Start the installation.

For example:

- `./pctAIX.bin -options my_options_file.txt -silent`
- `./pctHPUX.bin -options my_options_file.txt -silent`
- 64-bit platforms: `./pctHPUXIA64.bin -options my_options_file.txt -silent`

- `./pctLinux.bin -options my_options_file.txt -silent`
  - 64-bit platforms: `./pct.bin -options my_options_file.txt -silent`
  - Power platforms: `./pctLinuxPPC.bin -options my_options_file.txt -silent`
  - S/390 platforms: `./pctLinux390.bin -options my_options_file.txt -silent`
  - `./pctSolaris.bin -options my_options_file.txt -silent`
  -  `pctWindows.exe -options my_options_file.txt -silent`
  -  64-bit platforms: `pctWindowsIA64.exe -options my_options_file.txt -silent`
5. After using the Profile creation wizard, examine the logs for success.

## Logging

The following log files record information about profile creation:

- The `install_root/logs/log.txt` file records installation status.
- The `install_root/profiles/profile_name/logs/pctLog.txt` file records installation events that occur when creating profiles with the Profile creation wizard.
- The `install_root/logs/wasprofile/wasprofile_create_profile_name.log` file records installation events that occur when creating profiles.
- The `install_root/logs/wasprofile/wasprofile_delete_profile_name.log` file records installation events that occur when deleting profiles.

See the *Troubleshooting and support* PDF for more information.

## Usage notes

- The file is not a read-only file.
- Edit this file directly with your flat file editor of choice, such as WordPad on a Windows platform.
- The file is updated when you specify the `-options` parameter when using the Profile creation wizard and the file does not yet exist.
- The file must exist to perform a silent installation. The installation program reads this file to determine installation option values when you install silently.
- Save the file in a location that you can identify when you specify the fully qualified path as part of the profile creation command.

## Naming considerations

Consider the following recommendations when supplying names for the profile and other objects.

### Naming the profile

Use the following guidelines when supplying a value for the profile name directive:

```
-W profilenamepanelInstallWizardBean.profileName
```

**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as `*&?`
- Slashes (`/`) or (`\`)

Double-byte characters are allowed.

### Avoiding reserved names

Avoid the following reserved folder names as values for the directives in the responsefile.pct.NDstandAloneProfile.txt file and in the responsefile.pct.NDmanagedProfile.txt file:

```
-W nodehostnamepanelInstallWizardBean.nodeName=""  
-W nodehostnamepanelInstallWizardBean.hostName=""  
-W setnondmgrcellnameinglobalconstantsInstallWizardBean.value=""
```

Avoid the following reserved folder names as values for the directives in the responsefile.pct.NDdmgrProfile.txt file:

```
-W nodehostandcellnamepanelInstallWizardBean.nodeName=""  
-W nodehostandcellnamepanelInstallWizardBean.hostName=""  
-W nodehostandcellnamepanelInstallWizardBean.cellName=
```

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

### Node and cell name considerations

**Windows** The profiles directory path must be no longer than 80 characters.

### Host name considerations

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the hostName property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as xmachine.manhattan.ibm.com
- The default short DNS host name string, such as xmachine
- Numeric IP address, such as 127.1.255.3

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected

from the network. Define the short name to 127.0.0.1 (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.

A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A format disadvantage is that the numeric IP address is fixed. You must change the setting of the hostName property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

### Example responsefile.pct.NDdmgrProfile.txt file

```
#####
#
# Response file for Websphere Application Server 6.0 dmgr profile creation
#
# This options file is located in the CD_ROOT\WAS\ directory and in the
# install_root\bin\ProfileCreator directory.
#
# To use the options file under CD_ROOT\WAS\ directory, follow the instructions
# in CD_ROOT\WAS\responsefile.nd.txt. The WebSphere Application Server
# Network Deployment installer locates this file during silent installation
# and automatically runs the silent profile creation at the end of installation.
#
# To use the options file under install_root\bin\ProfileCreator for silent profile
# creation, you must change various values in the file and use the following
# command line arguments:
#
# -options "responsefile.pct.NDdmgrProfile.txt" -silent
#
#####

#####
#
# Profile name
#
# Set the profile name for installing a deployment manager profile. The profile
# name must be unique for this WebSphere Application Server installation.
#
-W profilenamepanelInstallWizardBean.profileName="profileDmgr"

#####
#
# If you want to set this profile to be your default profile, set to "true".
# Otherwise set to "false". If this is the first profile being created, the profile
# automatically is the default.
#
-W profilenamepanelInstallWizardBean.isDefault="false"

#####
#
# Profile location
#
# Specify a directory to contain the files that define the run-time environment,
# such as commands, configuration files, and log files. If the directory contains
# spaces, enclose it in double-quotes as shown in the Windows example below.
#
# Note that spaces in the install location is only supported on Windows
# operating systems.
#
# Default Install Location:
#
# -P installLocation="<WAS_HOME>\profiles\<PROFILE_NAME>"
```

```

#
-P installLocation="C:\Program Files\IBM\WebSphere\AppServer\profiles\profileDmgr"

#####
#
# Node name
#
# Please select the node name for the Application Server. Node name under one cell
# has to be unique.
#
# Replace YOUR_NODE_NAME with the actual node name.
#
-W nodehostandcellnamepanelInstallWizardBean.nodeName="YOUR_NODE_NAME"

#####
#
# Host name
#
# Specify the host name for the Application Server. The host name is the domain
# name system (DNS) name (short or long) or the IP address of this computer.
#
# Replace YOUR_HOST_NAME with the actual host name. Comment the line to use
# the default value.
#
-W nodehostandcellnamepanelInstallWizardBean.hostName="YOUR_HOST_NAME"

#####
#
# Cell name
#
# Specify the cell name for the Application Server.
#
# If you plan to migrate a V5 deployment manager cell to this V6 deployment
# manager, specify the same cell name as the V5 cell.
#
# Replace YOUR_CELL_NAME with the actual cell name.
#
-W nodehostandcellnamepanelInstallWizardBean.cellName="YOUR_CELL_NAME"

#####
#
# Port value assignment
#
# The following entries are used to reset port numbers used in the configuration
#
# They are currently set to the defaults.
# Please check to make sure there are no Port Conflicts.
# Port numbrers for each profile can be found in:
# <profile>/config/cells/<cell name>/nodes/<node name>/serverindex.xml
#
-W pctdmgrprofileportspanelInstallWizardBean.WC_adminhost="9060"
-W pctdmgrprofileportspanelInstallWizardBean.WC_adminhost_secure="9043"
-W pctdmgrprofileportspanelInstallWizardBean.BOOTSTRAP_ADDRESS="9809"
-W pctdmgrprofileportspanelInstallWizardBean.SOAP_CONNECTOR_ADDRESS="8879"
-W pctdmgrprofileportspanelInstallWizardBean.SAS_SSL_SERVERAUTH_LISTENER_ADDRESS="9404"
-W pctdmgrprofileportspanelInstallWizardBean.CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS="9406"
-W pctdmgrprofileportspanelInstallWizardBean.CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS="9405"
-W pctdmgrprofileportspanelInstallWizardBean.ORB_LISTENER_ADDRESS="9101"
-W pctdmgrprofileportspanelInstallWizardBean.CELL_DISCOVERY_ADDRESS="7277"
-W pctdmgrprofileportspanelInstallWizardBean.DCS_UNICAST_ADDRESS="9352"

#####
#
# Windows service

```

```

#
# The following directives are to install services for WebSphere Application Server
# on Windows.
# Using Services, you can start and stop services,
# and configure startup and recovery actions.
# Set winServiceQuery="false" will turn off the function on windows system.
# You can ignore these or comment them out for other Operating Systems.
#
-W winservicepanelInstallWizardBean.winServiceQuery="true"

#####
# Specify account type of the service. Legal values are:
#
#   localsystem   - Indicates that you choose to use Local System account.
#   specifieduser - Indicates that you choose to use specified user account.
#
-W winservicepanelInstallWizardBean.accountType="localsystem"

#####
# If you chose to install a service above with the accountType="localsystem",
# the userName and password below can be ignored. If the accountType was set to:
# accountType="specifieduser", then you must specify the User Name and Password
# which are required to install the Services. The current user must be admin or must
# have admin authority to install a Service. Also the username
# which is given here must have "Log On as a Service " authority
# for the service to run properly.
#
# Replace YOUR_USER_NAME with your username.
#
-W winservicepanelInstallWizardBean.userName="YOUR_USER_NAME"

#####
# Replace YOUR_PASSWORD with your valid password.
#
-W winservicepanelInstallWizardBean.password="YOUR_PASSWORD"

#####
# Set the startup type of the WebSphere Application Server on Windows.
# Valid values are "automatic", "manual", and "disabled".
#
-W winservicepanelInstallWizardBean.startupType="manual"

#####
# Profile type
#
# Must be set to "dmgr" for installing a deployment manager Profile.
# Do not change this!
#
-W profilenamepanelInstallWizardBean.selection="dmgr"

```

## Using the Profile creation wizard to create a custom profile

This topic describes creating a run-time environment for a custom profile.

Before using the Profile creation wizard, install the core product files.

The Profile creation wizard is the wizard interface to the profile creation tool, wasprofile. See the description of the “wasprofile command” on page 86 for more information.

An error can occur when you have not provided enough system temporary space to create a profile. Verify that you have a minimum of 40 MB of temp space available before creating a profile.

You must have 30 MB of available disk space in the directory where you create a deployment manager profile.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

You must have 10 MB of available disk space in the directory where you create a custom profile.

Manually verify that the required space for creating a profile is available on AIX. A known problem in the underlying InstallShield for Multiplatforms (ISMP) code prevents proper space checking on AIX systems at the time that the product disc was created.

After installing the core product files for the Network Deployment product, you must create one of the following profiles to have an operational run-time environment:

- A custom profile.

This topic describes creating a custom profile. A custom profile is an empty node that you can customize to include application servers, clusters, or other Java processes, such as a messaging server.

You must have 10 MB of available disk space in the directory where you create a custom profile.

- A deployment manager.

See “Using the Profile creation wizard to create a deployment manager” on page 49. The deployment manager provides a single administrative interface to a logical group of application servers on one or more machines.

- An application server profile.

See “Using the Profile creation wizard to create an application server” on page 73. An Application Server profile has a default server (which is server1), the default application that includes the snoop servlet and the hitcount servlet, and application Samples. You can federate the Application Server or use it as a stand-alone Application Server.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

This topic describes creating a custom profile using the Profile creation wizard. You can use the Profile creation wizard in silent mode with a response file instead of the graphical user interface. See “responsefile.pct.NDmanagedProfile.txt” on page 65 for examples of using the Profile creation wizard in silent mode.

You can also use the **wasprofile** command to create a custom profile. See the description of the “wasprofile command” on page 86 for more information.

After creating a custom profile, you must have access to a running deployment manager to federate the node. Federating the custom profile makes the node operational. If the custom profile is on a machine that does not have a deployment manager, the deployment manager must be accessible over the network to allow the federation of the node.

You can federate the custom node as you create the custom profile, either with the Profile creation wizard or when using the **wasprofile** command. If you choose to federate, but the deployment manager is not running, the custom node is not usable. You must then either delete the profile directory or move the directory out of the profiles repository (profiles installation root directory) before creating another profile with the same name.



1. Install the product to create the core product files.
2. Start the Profile creation wizard to create a new run-time environment.

Several ways exist to start the wizard. The initial way to start the wizard is at the end of installation by selecting the check box to launch the Profile creation wizard.



One way to start the wizard is to issue the command directly from a command line.

The command is in the *install\_root/bin/ProfileCreator* directory. The name of the command varies per platform:


- `pctAIX.bin`
- `pctHPUX.bin`
- 64-bit platforms: `pctHPUXIA64.bin`
- `pctLinux.bin`
- 64-bit platforms: `pctLinuxIA64.bin` S/390 platforms: `pctLinux390.bin`
- Power platforms: `pctLinuxPPC.bin`
- `pctSolaris.bin`
-  `pctWindows.exe`
-  64-bit platforms: `pctWindowsIA64.exe`

Another way to start the Profile creation wizard is to select the wizard from the First steps console.



a. Open a command window.

b. Change directories to the `firststeps` directory in the installation root directory:

The installation root varies by platform:

- `./usr/IBM/WebSphere/AppServer/firststeps`
- `.../opt/IBM/WebSphere/AppServer/firststeps`
-  `C:\Program Files\IBM\WebSphere\AppServer\firststeps`

c. Issue the **firststeps** command to start the console:

-  `./firststeps.sh`
-  `firststeps.bat`

d. Select the Profile creation wizard option on the console.

The Profile creation wizard is an InstallShield for Multiplatforms application. The wizard loads the Java 2 SDK and then displays its Welcome panel.

See the description of the “firststeps command” on page 40 for more information.

3. Click **Next** on the Welcome panel.

The wizard displays the Profile type selection panel.

4. Select **Create a custom profile** and click **Next**.

The wizard displays the Custom-profile federation panel.

5. Specify the host name and SOAP port of the deployment manager and click **Next**.

After federation, the process in the custom profile is the nodeagent process. The nodeagent process is the agent of the deployment manager for the custom node. The nodeagent responds to commands from the deployment manager to perform tasks that include the following actions:

- Creating application server processes, clusters, and cluster members
- Starting and stopping application server processes
- Synchronizing configurations between the current edition on the deployment manager and the copy that exists on the node
- Deleting application server processes

See the system administration section of the information center for more information about node agents and their tasks.

### Should you federate the node?

Federate the custom node at this time if the deployment manager is running. Select the check box to federate the node at a later time if the deployment manager is not running.

If you are unsure whether the deployment manager is running, do not federate now. Federate the node later.



If security is enabled on the deployment manager node, you must federate later using the **addNode** command to enter a user ID and password on the command.

A possibility exists that the deployment manager is reconfigured to use the non-default remote method invocation (RMI) as the preferred Java Management Extensions (JMX) connector. (Click **System Administration > Deployment manager > Administrative services** in the administrative console of the deployment manager to verify the preferred connector type.)

If RMI is the preferred JMX connector, you must use the **addNode** command to federate the custom profile at a later time. Use the **addNode** command so that you can specify the JMX connector type and the RMI port.

If the deployment manager uses the default SOAP JMX connector type, specify the host name and SOAP port and federate the node now to create a functional node that you can customize.

### **Federating when the deployment manager is not available**

If you federate a custom node when the deployment manager is not running or is not available because of security being enabled or for other reasons, the installation indicator in the logs is INSTCONFFAIL to indicate a complete failure. The resulting custom profile is unusable. You must move the custom profile directory out of the profile repository (the profiles installation root directory) before creating another custom profile with the same profile name.

Click **Next** to display the Profile name panel.

6. Specify a name for the profile, then click **Next**.

**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as \* & ?
- Slashes (/) or (\)

Double-byte characters are allowed.

### **The default profile**

The first profile that you create on a machine is the default profile. The default profile is the default target for commands issued from the `bin` directory in the product installation root. When only one profile exists on a machine, every command works on the only server process in the configuration.

### **Addressing a profile in a multi-profile environment**

When two or more profiles exist on a machine, certain commands require that you specify the profile to which the command applies. These commands use the `-profileName` parameter to identify which profile to address. You might find it easier to use the commands that in the `bin` directory of each profile.

A command in the `profiles/profile_name/bin` directory has two lines. The first line sets the `WAS_USER_SCRIPT` environment variable for the command window. The variable sets up the command environment to address the profile. The second line calls the actual command in the `install_root/bin` directory.

The actual command queries the command shell to determine the calling profile and to autonomically address the command to the calling profile.

The wizard then displays the Profile directory panel.

7. Specify a location for the profile and click **Next**.

If you click **Back** and change the name of the profile, you must manually change the name on this panel when it displays again.

The wizard displays the Node and host names panel.

8. Specify the node and host characteristics for the custom profile and click **Next**.

### **Migration considerations**

If you plan to migrate an installation of V5.x Network Deployment to V6, use the same cell name for the V6 deployment manager as you used for the V5.x cell.

After migrating the cell, the V5 managed nodes are now managed by the V6 deployment manager in compatibility mode. You can migrate individual V5 managed nodes in the cell to V6. To do so, you must create a V6 profile with the same node name as the V5 managed node.

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

The custom profile has the following characteristics:

Field name	Default value	Constraints	Description
Node name	The name of your machine, or a unique derivation of the machine name.	Avoid using the reserved words.  Use a unique name within the deployment manager cell.  If you plan to migrate a V5 managed node, use the same node name for this V6 custom profile.	The name is used for administration within the deployment manager cell to which the custom profile is added. Use a unique name within the deployment manager cell.  After migrating a V5 deployment manager cell to a V6 deployment manager, you can migrate the V5 custom profiles that are running in compatibility mode in the V6 deployment manager.
Host name	The DNS name of your machine.	The host name must be addressable through your network.	Use the actual DNS name or IP address of your machine to enable communication with your machine. See additional information about the host name that follows this table.

### Node name considerations

 The profiles directory path must be no longer than 80 characters.

### Host name considerations

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the `hostName` property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as `xmachine.manhattan.ibm.com`
- The default short DNS host name string, such as `xmachine`
- Numeric IP address, such as `127.1.255.3`

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan

to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected from the network. Define the short name to 127.0.0.1 (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.

A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A format disadvantage is that the numeric IP address is fixed. You must change the setting of the `hostName` property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

After specifying custom profile characteristics, the wizard displays the Port value assignment panel.

9. Specify port assignments that do not conflict for the custom profile and click **Next**.

When federating a custom profile, the **addNode** command uses non-conflicting ports. This means that you can take the default port assignments as you create the profile, and let the **addNode** command specify non-conflicting ports as you federate the node. Port assignments must be unique on a machine. application server processes on different machines can use the same port assignments without conflict.

After specifying non-conflicting port assignments, the wizard displays the Profile summary panel.

10. Click **Next** to create the custom profile or click **Back** to change the characteristics of the custom profile. The wizard displays a Status panel as the wizard creates the custom profile.

At the end of the installation, the wizard displays the Profile creation is complete panel.

11. Click **Finish** to exit the Profile creation wizard.

You can create a custom profile. The node within the profile is empty until you federate the node and use the deployment manager to customize the node.

Refer to the description of the “wasprofile command” on page 86 to learn about creating this type of profile using a command instead of a wizard.

Federate the node into the deployment manager cell if you have not already done so as you created the node. Then use the deployment manager to create an application server on the node. Then you are ready to deploy an application.

Deploy an application to get started!

### **responsefile.pct.NDmanagedProfile.txt**

This topic describes the response file for creating a custom profile. Federate the custom node into a running deployment manager cell to make the node operational.

Create a custom node using the with an options response file after logging on as root on a Linux or UNIX platform, or as a user that belongs to the administrator group on a Windows platform. Some steps of the installation procedure on a Windows platform require the user to belong to the administrator group and to have the advanced user rights *Act as part of the operating system* and *Log on as a service*.

The response file is shipped with default values.

A common use for an options file is to run the Installation wizard in silent mode, which is referred to as installing silently. The wizard reads the options file to determine responses and does not display the graphical user interface. Issue the following command to use a copy of the options file named `myresponsefile.txt` for a silent installation:

```
pctWindows.exe -options "myresponsefile.txt" -silent
```

## Federating the custom profile

Several directives in the file provide options for how the custom node is federated into the deployment manager cell:

- **-W `pctfederationpanelInstallWizardBean.federateLater`**  
Set this value to `true` if the deployment manager is not running or is not accessible for any of the reasons in the following description of federation.
- **-W `pctfederationpanelInstallWizardBean.hostname`**  
Specify a value that resolves to the system where the deployment manager is running. See the following description of host name considerations for more information.
- **-W `pctfederationpanelInstallWizardBean.port`**  
Specify the value of the deployment manager SOAP port. You must specify the correct value. An incorrect value prevents node federation and results in a total failure with an `INSTCONFFAILED` indicator. The default SOAP port for the deployment manager is 8879.

## Should you federate the node?

Federate the custom node if you can. However, if any of the parameters that you supply are faulty, you not only do not federate the node, you create a faulty custom profile. Federate the node at the time that you perform the silent creation of the node if, and only if, the deployment manager is running and is accessible at the host name that you specify and over the SOAP port that you specify.

If you are unsure whether the deployment manager is running, do not federate now. Federate the node later.

If security is enabled on the deployment manager node, you must federate later using the **`addNode`** command to enter a user ID and password on the command.

A possibility exists that the deployment manager is reconfigured to use the non-default remote method invocation (RMI) as the preferred Java Management Extensions (JMX) connector. (Click **System Administration > Deployment manager > Administrative services** in the administrative console of the deployment manager to verify the preferred connector type.)

If RMI is the preferred JMX connector, you must use the **`addNode`** command to federate the custom profile at a later time. Use the **`addNode`** command so that you can specify the JMX connector type and the RMI port.

If the deployment manager uses the default SOAP JMX connector type, specify the host name and SOAP port and federate the node now to create a functional node that you can customize.

## Federating when the deployment manager is not available

If you federate a custom node when the deployment manager is not running or is not available because of security being enabled or for other reasons, the installation indicator in the logs is `INSTCONFFAIL` to indicate a complete failure. The resulting custom profile is unusable. You must move the custom profile directory out of the profile repository (the profiles installation root directory) before creating another custom profile with the same profile name.

## Avoiding the use of the -silent option within the options response file

A problem occurs when the -silent option exists in the file. The file works with the option during a direct call to the profile creation wizard, but fails when called from a silent product installation. See the *Installing your application serving environment* PDF for information about creating a profile silently during a silent product installation.

The option is unnecessary. Avoid using the option to avoid problems.

## Response file locations

The example options response files are in two locations.

### Example files:

- responsefile.pct.NDdmgrProfile.txt
- responsefile.pct.NDmanagedProfile.txt
- responsefile.pct.NDstandAloneProfile.txt

### Location:

Table 6. Option response file locations

Product disc location	Installed location
/WAS directory	install_root/bin/ProfileCreator directory

Use the file on the product disc to install the Network Deployment product silently and create a profile.

After installing the Network Deployment product, you can use the installed response file with the -options parameter on the Profile creation wizard command.

## Required disk space

Profile	Required disk space	Required temp space
Deployment manager profile	30 MB	40 MB
Custom profile	10 MB	40 MB
Application server profile	200 MB	40 MB

## Creating an operational environment during product installation

Version 6 installation of the Network Deployment product is a two-step process:

1. Installing the core product files and feature files.
2. Creating a deployment manager profile, a custom profile, or an application server profile.

The sample options response file, responsefile.nd.txt, controls the first part of the installation and can also start the second part of the installation. To create a profile as part of installing the core product files, use the option in the responsefile.nd.txt file that identifies the response file for creating a profile. The profile response file lets you use the Profile creation wizard silently.

To edit and use the appropriate response file for creating a profile, perform the following procedure:

1. Copy the appropriate file from the WAS directory on the product disc to a place that you can easily identify on your machine. The example files are:

To create a:	Copy the following response file:
Deployment manager profile	responsefile.pct.NDmgrProfile.txt
Custom profile	responsefile.pct.NDmanagedProfile.txt
Application server profile	responsefile.pct.NDstandAloneProfile.txt

- Edit the file to customize the values for your installation.
- Verify that no `-silent` option exists in the response file for the Profile creation wizard. If the option exists, the profile is not created.
- Save the file.
- Edit the `responsefile.nd.txt` file to identify the location and name of the profile response file. Change the value of the `-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation` option to identify the file. For example:  

```
-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation=
"/opt/IBM/WebSphere/MyOptionFiles/customProfile.txt"
```
- Start the installation. For example:  

```
install -options "myresponsefile.txt" -silent
```
- After the installation, examine the logs for success.

### Creating a profile after installation

Version 6 installation of the Network Deployment product is a two-step process:

- Installing the core product files and feature files
- Creating a deployment manager profile, a custom profile, or an application server profile

When the core product files exist, create a profile at any time using the Profile creation wizard. Start the wizard from the First steps console or directly using the Profile creation wizard command.

You can also use one of the following sample options response files for profiles to create a profile silently using the Profile creation wizard in silent mode. To edit and use the appropriate response file for creating a profile, perform the following procedure:

- Copy the appropriate file from the `install_root/bin/ProfileCreator` directory to a place that you can easily identify on your machine. The example files are:



To create a profile for a:	Copy the following response file:
Deployment manager	responsefile.pct.NDmgrProfile.txt
Managed node	responsefile.pct.NDmanagedProfile.txt
Stand-alone application server	responsefile.pct.NDstandAloneProfile.txt

For example, copy the file as `my_options_file.txt`

- Edit the file to customize the values for your installation.
- Save the file.
- Start the installation.

For example:

- `./pctAIX.bin -options my_options_file.txt -silent`
- `./pctHPUX.bin -options my_options_file.txt -silent`
- 64-bit platforms: `./pctHPUXIA64.bin -options my_options_file.txt -silent`
- `./pctLinux.bin -options my_options_file.txt -silent`
- 64-bit platforms: `./pct.bin -options my_options_file.txt -silent`
- Power platforms: `./pctLinuxPPC.bin -options my_options_file.txt -silent`

- S/390 platforms: `./pctLinux390.bin -options my_options_file.txt -silent`
- `./pctSolaris.bin -options my_options_file.txt -silent`
-  `pctWindows.exe -options my_options_file.txt -silent`
-  64-bit platforms: `pctWindowsIA64.exe -options my_options_file.txt -silent`

5. After using the Profile creation wizard, examine the logs for success.

## Logging

The following log files record information about profile creation:

- The `install_root/logs/log.txt` file records installation status.
- The `install_root/profiles/profile_name/logs/pctLog.txt` file records installation events that occur when creating profiles with the Profile creation wizard.
- The `install_root/logs/wasprofile/wasprofile_create_profile_name.log` file records installation events that occur when creating profiles.
- The `install_root/logs/wasprofile/wasprofile_delete_profile_name.log` file records installation events that occur when deleting profiles.

See the *Troubleshooting and support* PDF for more information.

## Usage notes

- The file is not a read-only file.
- Edit this file directly with your flat file editor of choice, such as WordPad on a Windows platform.
- The file is updated when you specify the `-options` parameter when using the Profile creation wizard and the file does not yet exist.
- The file must exist to perform a silent installation. The installation program reads this file to determine installation option values when you install silently.
- Save the file in a location that you can identify when you specify the fully qualified path as part of the profile creation command.

## Naming considerations

Consider the following recommendations when supplying names for the profile and other objects.

### Naming the profile

Use the following guidelines when supplying a value for the profile name directive:

```
-W profilenamepanelInstallWizardBean.profileName
```

**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as `*&?`
- Slashes (`/`) or (`\`)

Double-byte characters are allowed.

### Avoiding reserved names

Avoid the following reserved folder names as values for the directives in the `responsefile.pct.NDstandAloneProfile.txt` file and in the `responsefile.pct.NDmanagedProfile.txt` file:



```
-W nodehostnamepanelInstallWizardBean.nodeName=""  
-W nodehostnamepanelInstallWizardBean.hostName=""  
-W setnondmgrcellnameinglobalconstantsInstallWizardBean.value=""
```

Avoid the following reserved folder names as values for the directives in the `responsefile.pct.NDmgrProfile.txt` file:

```
-W nodehostandcellnamepanelInstallWizardBean.nodeName=""  
-W nodehostandcellnamepanelInstallWizardBean.hostName=""  
-W nodehostandcellnamepanelInstallWizardBean.cellName=
```

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

### Node and cell name considerations

**Windows** The profiles directory path must be no longer than 80 characters.

### Host name considerations

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the `hostName` property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as `xmachine.manhattan.ibm.com`
- The default short DNS host name string, such as `xmachine`
- Numeric IP address, such as `127.1.255.3`

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected from the network. Define the short name to `127.0.0.1` (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.



A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A format disadvantage is that the numeric IP address is fixed. You must change the setting of the `hostName` property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

### Example responsefile.pct.NDmanagedProfile.txt file

**Tip:** A custom profile must be added into a deployment manager cell to become operational. Because of this strong dependency on being a managed node, the profile is often referred to as a *managed profile* or as a managed node.

Of course, until you federate the node into a cell, the node is not managed. Another thing to keep in mind is that any federated node is a managed node, including federated nodes within Application Server profiles.

The following response file refers to the term *managed* instead of the term *custom* in many directive names. Even so, all of the directives in this response file help to define a custom profile.

```
#####
#
# Response file for WebSphere Application Server v6.0 custom profile creation
#
# This options file is located in the CD_ROOT\WAS\ directory and in the
# install_root\bin\ProfileCreator directory.
#
# To use the options file under CD_ROOT\WAS\ directory, follow the instructions
# in CD_ROOT\WAS\responsefile.nd.txt. The WebSphere Application Server
# Network Deployment installer locates this file during silent installation
# and automatically runs the silent profile creation at the end of installation.
#
# To use the options file under install_root\bin\ProfileCreator for silent profile
# creation, you must change various values in the file and use the following command
# line arguments:
#
# -options "responsefile.pct.NDmanagedProfile.txt" -silent
#
#####

#####
#
# Profile name
#
# Set the name for this custom profile. The profile name must be unique for this
# WebSphere Application Server installation.
#
#
-W profilenamepanelInstallWizardBean.profileName="profileManaged"

#####
# If you want to set this profile to be your default profile, set to "true".
# Otherwise set to "false". If this is the first profile being created, the profile
# automatically is the default.
#
-W profilenamepanelInstallWizardBean.isDefault="false"

#####
#
# Profile location
#
# Specify a directory to contain the files that define the run-time environment,
```

```

# such as commands, configuration files, and log files. If the directory contains
# spaces, enclose it in double-quotes as shown in the Windows example below.
#
# Note that spaces in the install location is only supported on Windows
# operating systems.
#
# Default Install Location:
#
# -P installLocation="<WAS_HOME>\profiles\<PROFILE_NAME>"
#
-P installLocation="C:\Program Files\IBM\WebSphere\AppServer\profiles\profileManaged"

```

```

#####
#
# Node name
#
# Please select the node name for the Application Server. Node name under one cell
# has to be unique.
#
# If you plan to migrate a V5 deployment manager cell, the V5 managed nodes are also
# migrated to the V6 cell. To incrementally migrate an individual V5 managed node
# to V6, you must use the same node name for the V6 Application Server profile.
#
# Replace YOUR_NODE_NAME with the actual node name.
#
-W nodehostnamepanelInstallWizardBean.nodeName="YOUR_NODE_NAME"

```

```

#####
#
# Host name
#
# Specify the host name for the Application Server. The host name is the domain
# name system (DNS) name (short or long) or the IP address of this computer.
#
# Replace YOUR_HOST_NAME with the actual host name. Comment the line to use
# the default value.
#
-W nodehostnamepanelInstallWizardBean.hostName="YOUR_HOST_NAME"

```

```

#####
#
# Cell name
#
# You should not Modify this, unless absolutely necessary
#
# The Wizard would set this to short local host name + "Node##Cell" by default.
#
# If you would like to override the resolved cell name value, uncomment the line and
# replace YOUR_CELL_NAME with <YOUR_OWN_VALUE>
#
-W setnondmgrcellnameinglobalconstantsInstallWizardBean.value="YOUR_CELL_NAME"

```

```

#####
#
# Ports value assignment
#
# The following entries are used to reset port numbers used in the configuration
#
# They are currently set to the defaults.
# Please check to make sure there are no port conflicts.
# Port nubmers for each profile can be find in:
# <profile>/config/cells/<cell name>/nodes/<node name>/serverindex.xml
#

```

```

# If you specify true for the value of
# the -W pctfederationpanelInstallWizardBean.federateLater
# directive, port numbers are assigned automatically when you federate the
# node with the addNode command. The following port numbers do not apply.
#
-W pctmanagedprofileportspanelInstallWizardBean.BOOTSTRAP_ADDRESS="2809"
-W pctmanagedprofileportspanelInstallWizardBean.SOAP_CONNECTOR_ADDRESS="8878"
-W pctmanagedprofileportspanelInstallWizardBean.SAS_SSL_SERVERAUTH_LISTENER_ADDRESS="9901"
-W pctmanagedprofileportspanelInstallWizardBean.CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS="9201"
-W pctmanagedprofileportspanelInstallWizardBean.CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS="9202"
-W pctmanagedprofileportspanelInstallWizardBean.ORB_LISTENER_ADDRESS="9100"
-W pctmanagedprofileportspanelInstallWizardBean.NODE_DISCOVERY_ADDRESS="7272"
-W pctmanagedprofileportspanelInstallWizardBean.NODE_MULTICAST_DISCOVERY_ADDRESS="5000"
-W pctmanagedprofileportspanelInstallWizardBean.NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS="5001"
-W pctmanagedprofileportspanelInstallWizardBean.DCS_UNICAST_ADDRESS="9353"

```

```
#####
```

```

#
# Federation
#
# A custom profile contains an empty node that must be federated to a deployment
# manager to become a functional managed node. Identify a running deployment
# manager that will administer the node or choose to federate the node later
# using the addNode command.
#
# Set to "true" if you want to federate this custom node later using the addNode
# command. You must federate this node later if the deployment manager :
#     - is not running.
#     - has security enabled.
#     - has the SOAP connector disabled
#
# If you want to federate it now, set to "" and fill in the entries for the host
# and port of the deployment manager.
#
-W pctfederationpanelInstallWizardBean.federateLater=""

```

```
#####
```

```

# Specify the host name of the deployment manager for federation.
#
-W pctfederationpanelInstallWizardBean.hostname="YOUR_DEPLOYMENT_MANAGER_HOST_NAME"

```

```
#####
```

```

# Specify the port number where the deployment manager (DMGR) is reachable on the
# above host. The default port value is "8879".
#
-W pctfederationpanelInstallWizardBean.port="YOUR_DEPLOYMENT_MANAGER_PORT_NUMBER"

```

```
#####
```

```

#
# Profile type
#
# Must be set to "managed" for installing a custom profile. Do not change this!
#
-W profiletypepanelInstallWizardBean.selection="managed"

```

## Using the Profile creation wizard to create an application server

The Profile creation wizard can create an application server profile on any machine where the core product files exist.

Before using the Profile creation wizard, install the core product files.

The Profile creation wizard is the wizard interface to the profile creation tool, `wasprofile`. See the description of the “`wasprofile` command” on page 86 for more information.

An error can occur when you have not provided enough system temporary space to create a profile. Verify that you have a minimum of 40 MB of temp space available before creating a profile.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

**Important:**

You must have 30 MB of available disk space in the directory where you create a deployment manager profile.

You must have 10 MB of available disk space in the directory where you create a custom profile.

Manually verify that the required space for creating a profile is available on AIX. A known problem in the underlying InstallShield for Multiplatforms (ISMP) code prevents proper space checking on AIX systems at the time that the product disc was created.

**Important:**

After installing the core product files, you must create one of the following profiles to have an operational run-time environment:

- An application server that is described in this topic.  
An application server profile has a default server (which is `server1`), the default application that includes the snoop servlet and the hitcount servlet, and application Samples. You can federate the application server or use it as a stand-alone application server.
- A deployment manager.  
See “Using the Profile creation wizard to create a deployment manager” on page 49. The deployment manager provides a single administrative interface to a logical group of application servers on one or more machines.
- A custom profile that you must federate to make operational.  
See “Using the Profile creation wizard to create a custom profile” on page 60. A custom profile is an empty node that you can customize to include application servers, clusters, or other Java processes, such as a messaging server.

This procedure describes creating an application server profile using the graphical user interface provided by the Profile creation wizard.

You can use the Profile creation wizard in silent mode with a response file instead of a graphical user interface. See “`responsefile.pct.NDstandAloneProfile.txt`” on page 78 for examples of using the Profile creation wizard in silent mode.



You can also use the **`wasprofile`** command to create an application server profile. See the description of the “`wasprofile` command” on page 86 for more information.

1. Start the Profile creation wizard to create a new run-time environment.

Several ways exist to start the wizard. The initial way to start the wizard is at the end of installation by selecting the check box to launch the Profile creation wizard.

One way to start the wizard is to issue the command directly from a command line.


The command is in the `install_root/bin/ProfileCreator` directory. The name of the command varies per platform:

- `pctAIX.bin`
- `pctHPUX.bin`
- 64-bit platforms: `pctHPUXIA64.bin`
- `pctLinux.bin`
- 64-bit platforms: `pctLinux390.bin` S/390 platforms: `pctLinux390.bin`
- Power platforms: `pctLinuxPPC.bin`
- `pctSolaris.bin`
-  `pctWindows.exe`
-  64-bit platforms: `pctWindowsIA64.exe`



Another way to start the Profile creation wizard is to select the wizard from the First steps console.

- Open a command window.
- Change directories to the `firststeps` directory in the installation root directory:

The installation root varies by platform:

- `./usr/IBM/WebSphere/AppServer/firststeps`
- `.../opt/IBM/WebSphere/AppServer/firststeps`
-  `C:\Program Files\IBM\WebSphere\AppServer\firststeps`

- Issue the **firststeps** command to start the console:

-  `./firststeps.sh`
-  `firststeps.bat`

- Select the Profile creation wizard option on the console.

The Profile creation wizard is an InstallShield for Multiplatforms application. The wizard loads the Java 2 SDK and then displays its Welcome panel.

See the description of the “firststeps command” on page 40 for more information.

- Click **Next** on the Welcome panel.

The wizard displays the Profile type selection panel.

- Select the application server profile, then click **Next**.

The wizard displays the Profile directory panel.

- Specify a name for the profile, then click **Next**.

**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as `*&?`
- Slashes (`/`) or (`\`)

Double-byte characters are allowed.

### The default profile

The first profile that you create on a machine is the default profile. The default profile is the default target for commands issued from the `bin` directory in the product installation root. When only one profile exists on a machine, every command works on the only server process in the configuration.

### Addressing a profile in a multi-profile environment

When two or more profiles exist on a machine, certain commands require that you specify the profile to which the command applies. These commands use the `-profileName` parameter to identify which profile to address. You might find it easier to use the commands that in the `bin` directory of each profile.

A command in the `profiles/profile_name/bin` directory has two lines. The first line sets the `WAS_USER_SCRIPT` environment variable for the command window. The variable sets up the command environment to address the profile. The second line calls the actual command in the `install_root/bin` directory.

The actual command queries the command shell to determine the calling profile and to autonomically address the command to the calling profile.

The wizard then displays the Profile directory panel.

5. Accept the default directory or specify a non-default location, then click **Next**. Or click **Browse** to select a different location.

If you click **Back** and change the name of the profile, you must manually change the name on this panel when it displays again.

The wizard displays the Node and host name panel.

6. Specify the characteristics for the application server, then click **Next**.


Use unique names for each application server that you create.

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

Field name	Default value	Constraints	Description
Node name	Name of your machine	Avoid using the reserved words.	Pick any name you want. To help organize your installation, use a unique name if you plan to create more than one application server on the machine.
Host name	DNS name of your machine	Addressable through your network.	Use the actual DNS name or IP address of your machine to enable communication with your machine. See additional information about the host name following this table.

**Node name considerations:** If you plan to migrate an installation of V5.x Network Deployment to V6 and migrate one of the managed nodes in the cell, use the same node name for the V6 application server as you used for the V5.x managed node.

 The installation directory path must be no longer than 60 characters.

**Host name considerations:**

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the `hostName` property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as `xmachine.manhattan.ibm.com`
- The default short DNS host name string, such as `xmachine`
- Numeric IP address, such as `127.1.255.3`

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected from the network. Define the short name to `127.0.0.1` (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.

A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A format disadvantage is that the numeric IP address is fixed. You must change the setting of the `hostName` property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

After specifying application server characteristics, the wizard displays the Port value assignment panel.

7. Verify that the ports specified for the stand-alone application server are unique, then click **Next**.

**Windows** After specifying port assignments, the wizard displays the Windows service definition panel, if you are installing on a Windows platform.

8. **Windows** Choose whether to run the application server as a Windows service on a Windows platform and click **Next**.

Version 6 attempts to start Windows services for application server processes started by a **startServer** command. For example, if you configure an application server as a Windows service and issue the **startServer** command, the **wasservice** command attempts to start the defined service.

If you chose to install a local system service, you do not have to specify your user ID or password. If you create a specified user type of service, you must specify the user ID and the password for the user who is to run the service. The user must have *Log on as a service* authority for the service to run properly.

To perform this installation task, the user ID must not have spaces in its name. The ID must also belong to the administrator group and must have the advanced user rights *Act as part of the operating system* and *Log on as a service*. The Installation wizard grants the user ID the advanced user rights if it does not already have them, if the user ID belongs to the administrator group.

You can also create other Windows services after the installation is complete, to start other server processes. See “Automatically restarting server processes” on page 235 for more information.

The installation wizard shows which components are selected for installation in a pre-installation summary panel.

9. Click **Next** to create the application server or click **Back** to change the characteristics of the application server.

The wizard displays the Installation status panel that shows which components are installing.

When the installation is complete, the wizard displays the Profile creation is complete panel.

10. Click **Finish** to exit, then click **Profile creation wizard** on the First steps console to start the wizard again to create other application servers.



You can create an application server profile. The node within the profile has an application server named server1.

Refer to the description of the “wasprofile command” on page 86 to learn about creating this type of profile using a command instead of a wizard.

Deploy an application to get started!

### **responsefile.pct.NDstandAloneProfile.txt**

This topic describes the response file for creating a stand-alone application server profile.

Create a stand-alone application server profile with an options response file after logging on as root on a Linux or UNIX platform, or a user that belongs to the administrator group on a Windows platform. Some steps of the installation procedure on a Windows platform require the user to belong to the administrator group and to have the advanced user rights *Act as part of the operating system* and *Log on as a service*.

The response file is shipped with default values.

A common use for an options file is to run the Installation wizard in silent mode, which is referred to as installing silently. The wizard reads the options file to determine responses and does not display the graphical user interface. Issue the following command to use a copy of the options file named myresponsefile.txt for a silent installation:

```
pctWindows.exe -options "myresponsefile.txt" -silent
```

### **Avoiding the use of the -silent option within the options response file**

A problem occurs when the -silent option exists in the file. The file works with the option during a direct call to the profile creation wizard, but fails when called from a silent product installation. See the *Installing your application serving environment* PDF for information about creating a profile silently during a silent product installation.

The option is unnecessary. Avoid using the option to avoid problems.

### **Response file locations**

The example options response files are in two locations.

#### **Example files:**

- responsefile.pct.NDmgrProfile.txt
- responsefile.pct.NDmanagedProfile.txt
- responsefile.pct.NDstandAloneProfile.txt

#### **Location:**

*Table 7. Option response file locations*

<b>Product disc location</b>	<b>Installed location</b>
/WAS directory	install_root/bin/ProfileCreator directory

Use the file on the product disc to install the Network Deployment product silently and create a profile.

After installing the Network Deployment product, you can use the installed response file with the -options parameter on the Profile creation wizard command.

## Required disk space

Profile	Required disk space	Required temp space
Deployment manager profile	30 MB	40 MB
Custom profile	10 MB	40 MB
Application server profile	200 MB	40 MB

## Creating an operational environment during product installation

Version 6 installation of the Network Deployment product is a two-step process:

1. Installing the core product files and feature files.
2. Creating a deployment manager profile, a custom profile, or an application server profile.

The sample options response file, `responsefile.nd.txt`, controls the first part of the installation and can also start the second part of the installation. To create a profile as part of installing the core product files, use the option in the `responsefile.nd.txt` file that identifies the response file for creating a profile. The profile response file lets you use the Profile creation wizard silently.

To edit and use the appropriate response file for creating a profile, perform the following procedure:

1. Copy the appropriate file from the WAS directory on the product disc to a place that you can easily identify on your machine. The example files are:

To create a:	Copy the following response file:
Deployment manager profile	<code>responsefile.pct.NDdmgrProfile.txt</code>
Custom profile	<code>responsefile.pct.NDmanagedProfile.txt</code>
Application server profile	<code>responsefile.pct.NDstandAloneProfile.txt</code>

2. Edit the file to customize the values for your installation.
3. Verify that no `-silent` option exists in the response file for the Profile creation wizard. If the option exists, the profile is not created.
4. Save the file.
5. Edit the `responsefile.nd.txt` file to identify the location and name of the profile response file. Change the value of the `-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation` option to identify the file. For example:  

```
-W pctresponsefilelocationqueryactionInstallWizardBean.fileLocation=  
"/opt/IBM/WebSphere/MyOptionFiles/customProfile.txt"
```
6. Start the installation. For example:  

```
install -options "myresponsefile.txt" -silent
```
7. After the installation, examine the logs for success.

## Creating a profile after installation

Version 6 installation of the Network Deployment product is a two-step process:

1. Installing the core product files and feature files
2. Creating a deployment manager profile, a custom profile, or an application server profile

When the core product files exist, create a profile at any time using the Profile creation wizard. Start the wizard from the First steps console or directly using the Profile creation wizard command.

You can also use one of the following sample options response files for profiles to create a profile silently using the Profile creation wizard in silent mode. To edit and use the appropriate response file for creating a profile, perform the following procedure:



1. Copy the appropriate file from the `install_root/bin/ProfileCreator` directory to a place that you can easily identify on your machine. The example files are:

To create a profile for a:	Copy the following response file:
Deployment manager	responsefile.pct.NDmgrProfile.txt
Managed node	responsefile.pct.NDmanagedProfile.txt
Stand-alone application server	responsefile.pct.NDstandAloneProfile.txt

For example, copy the file as `my_options_file.txt`

2. Edit the file to customize the values for your installation.
3. Save the file.
4. Start the installation.

For example:

- `./pctAIX.bin -options my_options_file.txt -silent`
- `./pctHPUX.bin -options my_options_file.txt -silent`
- 64-bit platforms: `./pctHPUXIA64.bin -options my_options_file.txt -silent`
- `./pctLinux.bin -options my_options_file.txt -silent`
- 64-bit platforms: `./pct.bin -options my_options_file.txt -silent`
- Power platforms: `./pctLinuxPPC.bin -options my_options_file.txt -silent`
- S/390 platforms: `./pctLinux390.bin -options my_options_file.txt -silent`
- `./pctSolaris.bin -options my_options_file.txt -silent`
-  `pctWindows.exe -options my_options_file.txt -silent`
-  64-bit platforms: `pctWindowsIA64.exe -options my_options_file.txt -silent`

5. After using the Profile creation wizard, examine the logs for success.

## Logging

The following log files record information about profile creation:

- The `install_root/logs/log.txt` file records installation status.
- The `install_root/profiles/profile_name/logs/pctLog.txt` file records installation events that occur when creating profiles with the Profile creation wizard.
- The `install_root/logs/wasprofile/wasprofile_create_profile_name.log` file records installation events that occur when creating profiles.
- The `install_root/logs/wasprofile/wasprofile_delete_profile_name.log` file records installation events that occur when deleting profiles.

See the *Troubleshooting and support* PDF for more information.

## Usage notes

- The file is not a read-only file.
- Edit this file directly with your flat file editor of choice, such as WordPad on a Windows platform.
- The file is updated when you specify the `-options` parameter when using the Profile creation wizard and the file does not yet exist.
- The file must exist to perform a silent installation. The installation program reads this file to determine installation option values when you install silently.
- Save the file in a location that you can identify when you specify the fully qualified path as part of the profile creation command.

## Naming considerations

Consider the following recommendations when supplying names for the profile and other objects.

### Naming the profile

Use the following guidelines when supplying a value for the profile name directive:

```
-W profilenamepanelInstallWizardBean.profileName
```

**Profile naming guidelines:** The profile name can be any unique name with the following restrictions. Do not use any of the following characters when naming your profile:

- Spaces
- Illegal special characters that are not allowed within the name of a directory on your operating system, such as \*&?
- Slashes (/) or (\)

Double-byte characters are allowed.

### Avoiding reserved names

Avoid the following reserved folder names as values for the directives in the `responsefile.pct.NDstandAloneProfile.txt` file and in the `responsefile.pct.NDmanagedProfile.txt` file:

```
-W nodehostnamepanelInstallWizardBean.nodeName=""  
-W nodehostnamepanelInstallWizardBean.hostName=""  
-W setnondmgrcellnameinglobalconstantsInstallWizardBean.value=""
```

Avoid the following reserved folder names as values for the directives in the `responsefile.pct.NDdmgrProfile.txt` file:

```
-W nodehostandcellnamepanelInstallWizardBean.nodeName=""  
-W nodehostandcellnamepanelInstallWizardBean.hostName=""  
-W nodehostandcellnamepanelInstallWizardBean.cellName=
```

**Reserved names:** Avoid using reserved folder names as field values. The use of reserved folder names can cause unpredictable results. The following words are reserved:

- cells
- nodes
- servers
- clusters
- applications
- deployments

### Node and cell name considerations

**Windows** The profiles directory path must be no longer than 80 characters.

### Host name considerations

The host name is the network name for the physical machine on which the node is installed. The host name must resolve to a physical network node on the server. When multiple network cards exist in the server, the host name or IP address must resolve to one of the network cards. Remote nodes use the host name to connect to and to communicate with this node. Selecting a host name that other machines can reach within your network is extremely important. Do not use the generic localhost identifier for this value.

If you define coexisting nodes on the same computer with unique IP addresses, define each IP address in a domain name server (DNS) look-up table. Configuration files for stand-alone Application Servers do not provide domain name resolution for multiple IP addresses on a machine with a single network address.

The value that you specify for the host name is used as the value of the hostName property in configuration documents for the stand-alone Application Server. Specify the host name value in one of the following formats:

- Fully qualified domain name servers (DNS) host name string, such as xmachine.manhattan.ibm.com
- The default short DNS host name string, such as xmachine
- Numeric IP address, such as 127.1.255.3

The fully qualified DNS host name has the advantage of being totally unambiguous and also flexible. You have the flexibility of changing the actual IP address for the host system without having to change the Application Server configuration. This value for host name is particularly useful if you plan to change the IP address frequently when using Dynamic Host Configuration Protocol (DHCP) to assign IP addresses. A format disadvantage is being dependent on DNS. If DNS is not available, then connectivity is compromised.

The short host name is also dynamically resolvable. A short name format has the added ability of being redefined in the local hosts file so that the system can run the Application Server even when disconnected from the network. Define the short name to 127.0.0.1 (local loopback) in the hosts file to run disconnected. A format disadvantage is being dependent on DNS for remote access. If DNS is not available, then connectivity is compromised.

A numeric IP address has the advantage of not requiring name resolution through DNS. A remote node can connect to the node you name with a numeric IP address without DNS being available. A format disadvantage is that the numeric IP address is fixed. You must change the setting of the hostName property in Express configuration documents whenever you change the machine IP address. Therefore, do not use a numeric IP address if you use DHCP, or if you change IP addresses regularly. Another format disadvantage is that you cannot use the node if the host is disconnected from the network.

### Example responsefile.pct.NDstandAloneProfile.txt file

```
#####  
#  
# Response file for WebSphere Application Server v6.0 stand alone profile  
# creation  
#  
# This options file is located in the CD_ROOT\WAS\ directory and in the  
# install_root\bin\ProfileCreator directory.  
#  
# To use the options file under CD_ROOT\WAS\ directory, follow the instructions  
# in CD_ROOT\WAS\responsefile.nd.txt. The WebSphere Application Server  
# Network Deployment installer locates this file during silent installation  
# and automatically runs the silent profile creation at the end of installation.  
#  
# To use the options file under install_root\bin\ProfileCreator for silent profile  
# creation, you must change various values in the file and use the following  
# command line arguments:  
#  
# -options "responsefile.pct.NDstandAloneProfile.txt" -silent  
#  
#####  
  
#####  
#  
# Profile name  
#  
# Set the profile name for installing a stand alone profile. The profile  
# name must be unique for this WebSphere Application Server installation.
```

```

#
-W profilenamepanelInstallWizardBean.profileName="profileStandAlone"

#####
# If you want to set this profile to be your default profile, set to "true".
# Otherwise set to "false". If this is the first profile being created, the profile
# automatically is the default.
#
-W profilenamepanelInstallWizardBean.isDefault="false"

#####
#
# Profile location
#
# Specify a directory to contain the files that define the run-time environment,
# such as commands,configuration files, and log files. If the directory contains
# spaces, enclose it in double-quotes as shown in the Windows example below.
#
# Note that spaces in the install location is only supported on Windows
# operating systems.
#
# Default Install Location:
#
#   -P installLocation="<WAS_HOME>\profiles\<PROFILE_NAME>"
#
-P installLocation="C:\Program Files\IBM\WebSphere\AppServer\profiles\profileStandAlone"

#####
#
# Node name
#
# Please select the node name for the Application Server. Node name under one cell
# has to be unique.
#
# If you plan to migrate a V5 deployment manager cell, the V5 managed nodes are also
# migrated to the V6 cell. To incrementally migrate an individual V5 managed node
# to V6, you must use the same node name for the V6 Application Server profile.
#
# Replace YOUR_NODE_NAME with the actual node name.
#
-W nodehostnamepanelInstallWizardBean.nodeName="YOUR_NODE_NAME"

#####
#
# Host name
#
# Specify the host name for the Application Server. The host name is the domain
# name system (DNS) name (short or long) or the IP address of this computer.
#
# Replace YOUR_HOST_NAME with the actual host name. Comment the line to use
# the default value.
#
-W nodehostnamepanelInstallWizardBean.hostName="YOUR_HOST_NAME"

#####
#
# Cell name
#
# You should not Modify this, unless absolutely necessary.
#
# The Wizard would set this to short local host name + "Node##Cell" by default.
#
# If you would like to override the resolved cell name value, uncomment the line and
# replace YOUR_CELL_NAME with <YOUR_OWN_VALUE>.

```

```

#
# -W setnondmgrcellnameinglobalconstantsInstallWizardBean.value="YOUR_CELL_NAME"

#####
#
# Port value assignment
#
# The following entries are used to reset port numbers used in the configuration
#
# They are currently set to the defaults.
# Please check to make sure there are no Port Conflicts.
# Port numbers for each profile can be find in:
# <profile>/config/cells/<cell name>/nodes/<node name>/serverindex.xml
#
-W pctdefaultprofileportspanelInstallWizardBean.WC_defaulthost="9080"
-W pctdefaultprofileportspanelInstallWizardBean.WC_adminhost="9060"
-W pctdefaultprofileportspanelInstallWizardBean.WC_defaulthost_secure="9443"
-W pctdefaultprofileportspanelInstallWizardBean.WC_adminhost_secure="9043"
-W pctdefaultprofileportspanelInstallWizardBean.BootSTRAP_ADDRESS="2809"
-W pctdefaultprofileportspanelInstallWizardBean.SOAP_CONNECTOR_ADDRESS="8880"
-W pctdefaultprofileportspanelInstallWizardBean.SAS_SSL_SERVERAUTH_LISTENER_ADDRESS="9401"
-W pctdefaultprofileportspanelInstallWizardBean.CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS="9403"
-W pctdefaultprofileportspanelInstallWizardBean.CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS="9402"
-W pctdefaultprofileportspanelInstallWizardBean.ORB_LISTENER_ADDRESS="9100"
-W pctdefaultprofileportspanelInstallWizardBean.DCS_UNICAST_ADDRESS="9353"
-W pctdefaultprofileportspanelInstallWizardBean.SIB_ENDPOINT_ADDRESS="7276"
-W pctdefaultprofileportspanelInstallWizardBean.SIB_ENDPOINT_SECURE_ADDRESS="7286"
-W pctdefaultprofileportspanelInstallWizardBean.SIB_MQ_ENDPOINT_ADDRESS="5558"
-W pctdefaultprofileportspanelInstallWizardBean.SIB_MQ_ENDPOINT_SECURE_ADDRESS="5578"

#####
#
# Windows service
#
# The following directives are to install services for
# WebSphere Application Server on Windows.
# Using Services, you can start and stop services,
# and configure startup and recovery actions.
# Set winServiceQuery="false" will turn off the function on windows system.
# You can ignore these or comment them out for other Operating Systems.
#
-W winservicepanelInstallWizardBean.winServiceQuery="true"

#####
# Specify account type of the service. Legal values are:
#
#    localsystem    - Indicates that you choose to use Local System account.
#    specifieduser  - Indicates that you choose to use specified user account.
#
-W winservicepanelInstallWizardBean.accountType="localsystem"

#####
# If you chose to install a service above with the accountType="localsystem",
# the userName and password below can be ignored. If the accountType was set to:
# accountType="specifieduser", then you must specify the User Name and Password
# which are required to install the Services. The current user must be admin or must
# have admin authority to install a Service. Also the username
# which is given here must have "Log On as a Service " authority
# for the service to run properly.
#
# Replace YOUR_USER_NAME with your username.
#
-W winservicepanelInstallWizardBean.userName="YOUR_USER_NAME"

#####

```



```

# Replace YOUR_PASSWORD with your valid password.
#
-W winservicepanelInstallWizardBean.password="YOUR_PASSWORD"

#####
# Set the startup type of the WebSphere Application Server on Windows.
# Valid values are "automatic", "manual", and "disabled".
#
-W winservicepanelInstallWizardBean.startupType="manual"

#####
# Profile type
#
# This must be set to "default" for installing a stand alone profile
# Do not change this!
#
-W profilenamepanelInstallWizardBean.selection="default"

```

## Deleting a profile

This topic describes how to manually delete a profile.

Before using the manual procedure to remove a profile, try the **wasprofile** command with the **-delete** option. For example, issue one of the following commands:

```

> UNIX
./wasprofile.sh -delete
                -profileName profile_name | -profilePath profile_path

```

```

> Windows
wasprofile.bat -delete
                -profileName profile_name | -profilePath profile_path

```

See “wasprofile command” on page 86.

If the command does not work, use this procedure to delete the profile.

This procedure describes how to manually delete a profile when the **wasprofile -delete** command results in the following message:

```
INSTCONFFAILED: Cannot delete profile
```

1. Delete the *profiles\_install\_root/profile\_name* directory.
2. If the *install\_root/properties/profileRegistry.xml* file exists, edit the file in a flat-file editor to delete the entry for the profile, if the entry is present.

The entry resembles the following example:

```

<profile isDefault="true"
        name="BadProfile"
        path="E:\IBM\WebSphere\AppServer\profiles\BadProfile"
        template="E:\IBM\WebSphere\AppServer\profileTemplates\default"/>

```

3. **UNIX** Compare the two batch files, *install\_root/properties/fsdb/\_was\_profile\_default/default.sh* and *install\_root/properties/fsdb/bad\_profile\_name.sh*.  
If the files are identical, delete the *install\_root/properties/fsdb/\_was\_profile\_default* directory and the *install\_root/properties/fsdb/bad\_profile\_name.sh* file.  
If the files are not identical, delete only the *install\_root/properties/fsdb/bad\_profile\_name.sh* file.
4. **Windows** Compare the two batch files, *install\_root\properties\fsdb\\_was\_profile\_default\default.bat* and *install\_root\properties\fsdb\bad\_profile\_name.bat*.

If the files are identical, delete the `install_root\properties\fsdb\_was_profile_default` directory and the `install_root\properties\fsdb\bad_profile_name.bat` file.

If the files are not identical, delete only the `install_root\properties\fsdb\bad_profile_name.bat` file.

See the description of the “wasprofile command” to learn more about the command-line method of working with profiles.

See “Using the Profile creation wizard” on page 45 for more information about creating profiles with the Profile creation wizard.

---

## wasprofile command

The **wasprofile** command line tool creates all Application Server run-time environments in Version 6. The command creates a profile, which is the set of files that define the run-time environment for a deployment manager, a custom profile, or a stand-alone Application Server.

The **wasprofile** command is also referred to as the *profile creation tool*.

## Introduction to terms that describe Version 6 profiles

The **wasprofile** command creates the run-time environment for a WebSphere Application Server process in a set of files called a *profile*. The profile defines the run-time environment and includes all of the files that the server processes in the run-time environment can change. The *profile creation tool* and its graphical user interface, the *Profile creation wizard*, are the only ways to create run-time environments in V6.

The *Profile creation wizard* is an InstallShield for Multiplatforms (ISMP) application. You can use the wizard to enter most of the parameters that are described in this topic. Some parameters, however, require you to use the **wasprofile** command. You must use the **wasprofile** command to delete a profile, for instance, because the Profile creation wizard does not provide a deletion function.

However, the Profile creation wizard also performs tasks that the wasprofile command does not. For instance, the wizard can create a Windows service for each profile that it creates. It can also assign non-conflicting ports based on previous Version 6 port assignments.

## Core product files

The core product files are the shared product binaries. The binary files are shared by all profiles.

The directory structure for V6 has two major divisions of files in the installation root directory for the product:

- The core product files are shared product binary files that do not change unless you install a refresh pack, a fix pack, or an interim fix. Some log information is also updated.
- The profiles directory is the default directory for creating profiles. The configuration for every defined Application Server process is within the profiles directory unless you specify a new directory when you create a profile. These files change as often as you create a new profile, reconfigure an existing profile, or delete a profile.

All of the folders except for the profiles directory and a few others such as the logs directory and the properties directory do not change unless you install service fixes. The profiles directory, however, changes each time you add, change, or delete a profile. The profiles directory is the default repository for profiles. However, you can put a profile anywhere on the machine provided there is enough available disk space.

If you put a profile in another existing folder in the installation root directory, a risk exists that the profile might be affected by the installation of a service fix that applies maintenance to the folder. Use a directory outside of the installation root directory when using a directory other than the `profiles` directory for creating profiles.

## WebSphere Application Server profile

The **wasprofile** command line tool defines each Application Server instance of a Version 6 product.

You must run the wizard or the command line tool each time that you want to create a stand-alone Application Server. A need for more than one stand-alone Application Server on a machine is common.

Administration is greatly enhanced when using V6 profiles instead of multiple product installs. Not only is disk space saved, but updating the product is simplified when you only maintain a single set of product core files. Also, creating new profiles is faster and less prone to error than full product installs, allowing a developer to create new disposable profiles of the product for development and testing.

You can run the Profile creation wizard or the profile creation tool to create a new Application Server environment on the same machine as an existing one. Simply define unique characteristics (such as profile name and node name) for the new profile. Each profile has its own administrative console and administrative scripting interface. Each Application Server process shares all run-time scripts, libraries, the Software Development Kit, and other core product files.

## Profile types

Templates for each profile are located in the `was_home/profileTemplates` directory.

Within this directory are the `default` folder, the `dmgr` folder, and the `managed` folder. These are the paths that you indicate while using the **wasprofile** command with the `-templatePath` option.

For example: `-templatePath /usr/WebSphere/AppServer/profileTemplates/default`

The **wasprofile** command in the Network Deployment product can create the following types of profiles:

### Deployment manager profile

The basic function of the deployment manager is to deploy applications to a cell of Application Servers, which it manages. Each Application Server that belongs to the cell is referred to as a *managed node*.

### Application Server profile

The basic function of the Application Server is to serve applications to the Internet or to an intranet.

An important product feature for the Network Deployment product is the ability to scale up a stand-alone Application Server profile by adding the Application Server node into a deployment manager cell. Multiple Application Server processes in a cell can deploy an application that is in demand. You can also remove an Application Server node from a cell to return the node to the status of a stand-alone Application Server.

Each stand-alone Application Server has its own administrative console application, which you use to manage the Application Server. You can also use the `wsadmin` scripting facility to perform every function that is available in the administrative console application.

There is no node agent process for a stand-alone Application Server unless you decide to add the Application Server node to a deployment manager cell. Adding the Application Server to a cell is known as *federation*. Federation changes the stand-alone Application Server into a managed node. At that point, you use the administrative console of the deployment manager to manage the node. If you remove the node from the deployment manager cell, use the administrative console and the scripting interface of the stand-alone Application Server to manage the process.

## Custom profile

The basic function of this profile that belongs to a deployment manager cell is to serve applications to the Internet or to an intranet under the management of the deployment manager.

The deployment manager changes a custom profile to a managed node by adding the node into the cell. This is also true when you add an Application Server into a cell. When either node is added to a cell, the node becomes a managed node. The *nodeagent* process is then instantiated on the managed node. The node agent acts on behalf of the deployment manager to control Application Server processes on the managed node. The node agent can start or stop Application Servers, for example.

A deployment manager can create multiple Application Servers on a managed node so long as the node agent process is running. Processes on the managed node can include cluster members that the deployment manager uses to balance the work load for heavily used applications.

Use the administrative console of the deployment manager to control all of the nodes that the deployment manager manages. You can also use the wsadmin scripting facility of the deployment manager to control any of the managed nodes. A custom profile does not have its own administrative console or scripting interface. You cannot manage the node directly with the wsadmin scripting facility. You must use the administrative interface of the deployment manager to manage a managed node.

A custom profile does not include default applications or a default server as the Application Server profile does. A custom profile is an empty node. Add the node to the deployment manager cell. Then you can use the administrative interface of the deployment manager to customize the managed node by creating clusters and Application Servers.

## Installed file set

You decide where to install the files that define a profile. The default location is in the `profiles` directory in the installation root directory. But you can change the location on the Profile creation wizard or in a parameter when using the command line tool. For example, assume that you create two profiles on a Linux platform with host name `devhost1`. The profile directories resemble the following example if you do not relocate them:

```
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile02
```

Suppose that you specify a different directory, such as `/opt/profiles`, for the profile directory field in the wizard. The profile directories resemble the following example:

```
/opt/profiles/devhost1Profile01  
/opt/profiles/devhost1Profile02
```

The following directories exist within a profile. This example assumes that a profile named `devhost1Profile01` exists:

```
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/bin  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/config  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/etc  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/firststeps  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/installableApps  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/installedApps  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/installedConnectors  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/installedFilters  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/logs  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/properties  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/samples  
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/temp
```

```
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/tranlog
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01/wstemp
```

## The profile repository

The profile repository is the default location of profile-related metadata. The repository is the default location for new profiles, which is often referred to as the profiles installation root directory.

However, you can decide where to install a profile. The default location of the profile repository is the *install\_root/profiles* directory. In the earlier example, creating two profiles on a Linux platform with host name devhost1 results in the following example directories in the profile repository:

```
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile01
/opt/IBM/WebSphere/AppServer/profiles/devhost1Profile02
```

When you specify a directory, such as */opt/profiles*, the profiles are no longer in the default repository, which is not a problem. For example, the following locations are valid:

```
/opt/profiles/devhost1Profile01
/opt/profiles/devhost1Profile02
```

## Location of the command file

The command file is located in the *install\_root/bin* directory. The command file is a script named *wasprofile.sh* for Linux and UNIX platforms or *wasprofile.bat* for Windows platforms.

The Profile creation wizard is the graphical user interface to the command line tool. The file name of the command that calls the Profile creation wizard varies per operating system platform. See “Using the Profile creation wizard” on page 45 for more information.

## Logging

The **wasprofile** command creates a log for every profile that it creates. The logs are in the *install\_root/logs/wasprofile* directory. The files are named in this pattern:  
*wasprofile\_create\_profile\_name.log*.

The command also creates a log for every profile that it deletes. The logs are in the *install\_root/logs/wasprofile* directory. The files are named in this pattern:  
*wasprofile\_delete\_profile\_name.log*.

## Required disk space

Manually verify that the required space for creating a profile is available on AIX. A known problem in the underlying InstallShield for Multiplatforms (ISMP) code prevents proper space checking on AIX systems at the time that the product disc was created.

An error can occur when you have not provided enough system temporary space to create a profile. Verify that you have a minimum of 40 MB of temp space available before creating a profile.

You must have 200 MB of available disk space in the directory where you create an Application Server profile.

## Network Deployment

### Tip:

You must have 30 MB of available disk space in the directory where you create a deployment manager profile.

You must have 10 MB of available disk space in the directory where you create a custom profile.

After installing the core product files, you must create one of the following profiles to have an operational run-time environment:

- An application server that is described in this topic.

An application server profile has a default server (which is server1), the default application that includes the snoop servlet and the hitcount servlet, and application Samples. You can federate the application server or use it as a stand-alone application server.

- A deployment manager.

See “Using the Profile creation wizard to create a deployment manager” on page 49. The deployment manager provides a single administrative interface to a logical group of application servers on one or more machines.

- A custom profile that you must federate to make operational.

See “Using the Profile creation wizard to create a custom profile” on page 60. A custom profile is an empty node that you can customize to include application servers, clusters, or other Java processes, such as a messaging server.

## Concurrent profile creation

**Important:** Concurrent profile creation is not supported at this time for one set of core product files. Concurrent attempts to create profiles result in a warning about a profile creation already in progress.

## Entering lengthy commands on more than one line

The length of the **wasprofile** command can exceed the normal shell window limit for one line of 256 characters. If your command is longer than the limit, issue the command on multiple lines by ending a line with a backward slash, pressing **Enter**, and continuing the command on the next line.

For example, on a Solaris system, the following command requires input on multiple lines:

```
./wasprofile.sh \  
-create -profileName bladetcb6profile \  
-profilePath /usr/IBM/WebSphere/AppServer/profiles/bladetcb6profile \  
-templatePath /usr/WebSphere/AppServer/profileTemplates/default \  
-nodeName bladetcb6node \  
-cellName bladetcb6Cell \  
-hostName bladetcb6.rtp.raleigh.ibm.com
```

Omit the line continuation character from the last line to signal the end of the command to the operating system.

## wasprofile.sh command syntax

List existing profiles:

```
# ./wasprofile.sh -listProfiles  
[-debug]
```

Delete profiles:

```
# ./wasprofile.sh -delete  
-profileName profile_name | -profilePath profile_path  
[-debug]
```

Create new profiles:

```
wasprofile.sh -create  
-profileName profile_name  
-profilePath fully_qualified_profile_path  
-templatePath template_path  
-nodeName node_name  
-cellName cell_name
```

```

-hostName host_name
-server iSeries_server_name
[-dmgrHost host_name]
[-dmgrPort SOAP_port]
[-startingPort starting_port | -portsFile filepath]
-winserviceCheck true | false
-winserviceAccountType specifieduser | localsystem
-winserviceUserName yourusername
-winservicePassword yourpassword
-winserviceStartupType manual | automatic | disabled
[-debug]

```

Get name of existing profile from path:

```

# ./wasprofile.sh -getName
    -profilePath profile_path
    [-debug]

```

Get path of existing profile from name:

```

# ./wasprofile.sh -getPath
    -profileName profile_name
    [-debug]

```

Check the integrity of the profile registry:

```

# ./wasprofile.sh -validateRegistry
    [-debug]

```

Check the integrity of the profile registry, removing profiles that are not found:

```

# ./wasprofile.sh -validateAndUpdateRegistry
    [-backup file_name]
    [-debug]

```

## wasprofile.bat command syntax

List existing profiles:

```

wasprofile.bat -listProfiles
    [-debug]

```

Delete profiles:

```

wasprofile.bat -delete
    -profileName profile_name | -profilePath profile_path
    [-debug]

```

Create new profiles:

```

wasprofile.bat -create
    -profileName profile_name
    -profilePath fully_qualified_profile_path
    -templatePath template_path
    -nodeName node_name
    -cellName cell_name
    -hostName host_name
    -server iSeries_server_name
    [-dmgrHost host_name]
    [-dmgrPort SOAP_port]
    [-startingPort starting_port | -portsFile filepath]
    -winserviceCheck true | false
    -winserviceAccountType specifieduser | localsystem
    -winserviceUserName yourusername
    -winservicePassword yourpassword
    -winserviceStartupType manual | automatic | disabled
    [-debug]

```



When the `-startingPort` parameter is not used, the profile creation tool uses the default port settings specified in the `serverindex.xml` file.

Get name of existing profile from path:

```
wasprofile.bat -getName
                -profilePath fully_qualified_profile_path
                [-debug]
```

Get path of existing profile from name:

```
wasprofile.bat -getPath
                -profileName profile_name
                [-debug]
```

Check integrity of profile registry:

```
wasprofile.bat -validateRegistry
                [-debug]
```

Check integrity of profile registry, removing unfound profiles:

```
wasprofile.bat -validateAndUpdateRegistry
                [-backup file_name]
                [-debug]
```

## Parameters

Supported arguments include:

### **-augment**

Refreshes or augments the given profile using the template in the `templatePath` parameter.

### **-backup** *file\_name*

Backs up the profile registry file to a file with the file name specified.

### **-cellname** *file\_name*

Specifies the cell name of the profile.

### **-create**

Creates the profile.

### **-debug**

Turns on the debug function of the Ant utility, which the **wasprofile** command uses.

### **-delete**

Deletes the profile.

### **-dmgrHost** *host\_name*

Identifies the machine where the deployment manager is running. Specify this parameter and the `dmgrPort` parameter to federate a custom profile as it is created.

The host name can be the long or short DNS name or the IP address of the deployment manager machine.

Specifying this optional parameter directs the **wasprofile** command to attempt to federate the custom node into the deployment manager cell as it creates the custom profile. This parameter is ignored when creating a deployment manager profile or an Application Server profile.

### **Federating when the deployment manager is not available**

If you federate a custom node when the deployment manager is not running or is not available because of security being enabled or for other reasons, the installation indicator in the logs is `INSTCONFFAIL` to indicate a complete failure. The resulting custom profile is unusable. You must move the custom profile directory out of the profile repository (the profiles installation root directory) before creating another custom profile with the same profile name.

- dmgrPort** *port\_number*  
Identifies the SOAP port of the deployment manager. Specify this parameter and the dmgrHost parameter to federate a custom profile as it is created. The deployment manager must be running and accessible.  
  
If you have enabled security or changed the default JMX connector type, you cannot federate with the **wasprofile** command. Use the **addNode** command instead.
- getName**  
Gets the name for a profile registered at a given file system path. Requires the `-profilePath` parameter.
- getPath**  
Gets the file system location for a profile of a given name. Requires the `-profileName` parameter.
- hostName** *host\_name*  
Specifies the host name where you are creating the profile. This should match the host name that you specified during installation of the initial product.
- listProfiles**  
Lists all defined profiles.
- nodeName** *node\_name*  
Specifies the node name for the node that is created with the new profile. Use a unique value within the cell on the machine. Each profile that shares the same set of product binaries must have a unique node name.
- portsFile** *file\_path*  
An optional parameter that specifies the path to a file that defines port settings for the new profile. When omitted, the wasprofile tool looks for the `install_root/profileTemplates/profile_type/actions/portsUpdate/bin/portdef.props` file.  
  
Do not use this parameter when using the startingPort parameter.
- profileName** *profile\_name*  
Specifies the name of the profile. Use a unique value when creating a profile. Each profile that shares the same set of product binaries must have a unique name.
- profilePath** *profile\_path*  
Specifies the fully qualified path to the profile.  
  
**Windows** Specify the full path to avoid an ANT scripting limitation that can cause a failure when federating the profile into a cell. For example:  
`-profilePath C:\IBM\WebSphere\AppServer\profiles\profile01`  
  
**Windows** If the fully qualified path contains spaces, enclose the value in quotation marks.
- startingPort** *startingPort*  
Specifies the starting port number for generating all ports for the profile. If not specified, the **wasprofile** command uses default ports specified in the `serverindex.xml` file.
- templatePath** *template\_path*  
Specifies the path to the templates in the shared binaries.
- validateAndUpdateRegistry** *registry\_file backup\_file*  
Checks all of the profiles that are listed in the profile registry to see if the profiles are present on the file system. Removes any missing profiles from the registry. Returns a list of the missing profiles that were deleted from the profile.
- validateRegistry** *registry\_file*  
Checks all of the profiles that are listed in the profile registry to see if the profiles are present on the file system. Returns a list of missing profiles.

**Windows** **-winserviceAccountType** *type\_of\_owner\_account*

The type of the owner account of the Windows service created for the profile can be either `specifieduser` or `localsystem`. The Windows service can run under the local account of the user who is creating the profile.

**Windows** **winserviceCheck** *value*

The value can be either `true` or `false`. Specify `true` to create a Windows service for the server process that is created within the profile. Specify `false` to not create the Windows service.

**Windows** **-winservicePassword** *yourpassword*

Specify the password for the specified user or the local account that is to own the Windows service.

**Windows** **-winserviceStartupType** *startup\_type*

Possible `startup_type` values are:

- `manual`
- `automatic`
- `disabled`

See “WASService command” on page 239 for more information about Windows services.

**Windows** **-winserviceUserName** *user\_ID*

Specify your user ID so that Windows can verify you as an ID that is capable of creating a Windows service. Your user ID must belong to the administrator group and have the following advanced user rights, *Act as part of the operating system* and *Log on as a service*

## Use case scenarios

Use cases are a description of common tasks for which the tool is used.

### Scenario: Creating a deployment manager profile on a Linux or UNIX platform

To create a deployment manager profile for user `shasti`:

```
wasprofile.sh -create
               -profileName shasti
               -profilePath ~/shasti/WebSphere
               -templatePath /opt/IBM/WebSphere/AppServer/profileTemplates/dmgr
               -cellName shaix1
               -hostName planetaix
               -nodeName dmgr1
```

On an Express product or a base product installation, the command does not create anything because the deployment manager template is not present.

On a Network Deployment product installation, the command creates a deployment manager profile named `shasti` in a cell named `shaix1` in location `~/shasti/WebSphere` with a node name of `dmgr1`.

### Scenario: Creating a deployment manager profile on a Windows platform

To create a server process for user `shasti`:

```
wasprofile.sh -create
               -profileName shasti
               -profilePath G:\shasti\WebSphere
               -templatePath C:\IBM\WebSphere\AppServer\profileTemplates\dmgr
               -cellName shwindows1
               -hostName planetnt
               -nodeName dmgr1
```

On an Express installation or on a base WebSphere Application Server product installation, the command does not create a deployment manager profile because the `dmgr` template does not exist in either product.

On a Network Deployment product installation, the command creates a deployment manager profile named `shasti` in a cell named `shwindows1` in location `G:\shasti\WebSphere` with a node name of `dmgr1`.

## Scenario: Creating a deployment manager profile in a multiuser environment on a Linux or UNIX platform

Follow these steps to create a server process for user shasti in a multiuser environment:

1. Create the server process:

```
wasprofile.sh -create
               -profileName shasti
               -profilePath ~/shasti/WebSphere
               -templatePath /opt/IBM/WebSphere/AppServer/profileTemplates/dmgr
               -cellName shaix1
               -hostName myhost
               -nodeName dmgr1
               -startingPort 12000
```

2. Change the owner of the folder:

```
chown -R shasti ~/shasti2/WebSphere
```

3. As a convenience, add a call to script ~/shasti/WebSphere/bin/setupCmdLine.sh in the profile of user shasti to set the environment when user shasti logs in.

4. Give these folder permissions to user shasti:

```
install_root/bin          --- rx (read and execute)
install_root/java         --- rx
install_root/properties   ----r (read)
install_root/deploytool   ----r
install_root/config       ----r
install_root/lib          ----r
install_root/classes      ----r
install_root/null         ----r
install_root/samples      ----r
install_root/Web          ----r
```

## Scenario: Deleting a profile

The following command is on more than one line for clarity. Enter the command on one line to delete the profile named shasti:

```
wasprofile.sh -delete
               -profileName shasti
```

## Scenario: Using predefined port numbers

When you use the wasprofile tool without the -startingPort parameter, the tool uses the /profileTemplates/profile\_type /actions/portsUpdate/bin/portdef.props file to set the initial ports.

### Example of using the -portsFile parameter

Copy the file, edit the port settings, and use your copy by using the -portsFile parameter as shown in the following example:

```
wasprofile.bat
  -create
  -profileName Wow_Profile
  -profilePath
    C:\ExpressV6\IBM\WebSphere\AppServer\profiles\Wow_Profile
  -templatePath
    C:\ExpressV6\IBM\WebSphere\AppServer\profileTemplates\default
  -nodeName Wow_node
  -cellName Wow_cell
  -hostName loyAllen
  -portsFile C:\temp\ports\portdef.props
```

Suppose that the portdef.props file has the following values:

```
WC_defaultHost=39080
WC_adminHost=39060
WC_defaultHost_secure=39443
```

```

WC_adminhost_secure=39043
BOOTSTRAP_ADDRESS=32809
SOAP_CONNECTOR_ADDRESS=38880
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=39401
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=39403
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=39402
ORB_LISTENER_ADDRESS=39100
DCS_UNICAST_ADDRESS=39353
SIB_ENDPOINT_ADDRESS=37276
SIB_ENDPOINT_SECURE_ADDRESS=37286
SIB_MQ_ENDPOINT_ADDRESS=35558
SIB_MQ_ENDPOINT_SECURE_ADDRESS=35578

```

As you run the command, messages similar to the following appear in the output stream:

```

replaceRegExpAllInstancesOfGivenTokenWithGivenValueForTheGivenFile:
[echo] File C:\ExpressV6\IBM\WebSphere\AppServer\profiles\
Wow_Profile/config/templates/default/serverentry-template.xml:
setting CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS to 39403

```

```

...
replaceRegExpAllInstancesOfGivenTokenWithGivenValueForTheGivenFile:
[echo] File C:\ExpressV6\IBM\WebSphere\AppServer\profiles\
Wow_Profile/config/templates/default/serverentry-template.xml:
setting CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS to 39402
...

```

The resulting serverindex.xml file looks similar to the following example:

```

<?xml version="1.0" encoding="UTF-8"?>
<serverindex:ServerIndex xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
...
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="BOOTSTRAP_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="32809"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="SOAP_CONNECTOR_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="38880"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="SAS_SSL_SERVERAUTH_LISTENER_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="39401"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="39403"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="39402"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="WC_adminhost">
    <endPoint xmi:id="EndPoint_..." host="*" port="39060"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="WC_defaulthost">
    <endPoint xmi:id="EndPoint_..." host="*" port="39080"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="DCS_UNICAST_ADDRESS">
    <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="39353"/>
  </specialEndpoints>
  <specialEndpoints xmi:id="NamedEndPoint_..."
                    endPointName="WC_adminhost_secure">
    <endPoint xmi:id="EndPoint_..." host="*" port="39043"/>
  </specialEndpoints>

```

```

<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="WC_defaulthost_secure">
  <endPoint xmi:id="EndPoint_..." host="*" port="39443"/>
</specialEndpoints>
<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="SIB_ENDPOINT_ADDRESS">
  <endPoint xmi:id="EndPoint_..." host="*" port="37276"/>
</specialEndpoints>
<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="SIB_ENDPOINT_SECURE_ADDRESS">
  <endPoint xmi:id="EndPoint_..." host="*" port="37286"/>
</specialEndpoints>
<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="SIB_MQ_ENDPOINT_ADDRESS">
  <endPoint xmi:id="EndPoint_..." host="*" port="35558"/>
</specialEndpoints>
<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="SIB_MQ_ENDPOINT_SECURE_ADDRESS">
  <endPoint xmi:id="EndPoint_..." host="*" port="35578"/>
</specialEndpoints>
<specialEndpoints xmi:id="NamedEndPoint_..."
    endPointName="ORB_LISTENER_ADDRESS">
  <endPoint xmi:id="EndPoint_..." host="IBMmachine" port="39100"/>
</specialEndpoints>
</serverEntries>
</serverindex:ServerIndex>

```

The **wasprofile** command creates a copy of the current portdefs.props file in the *install\_root\profiles\profile\_name\logs* directory.

Do not use the portsFile parameter when using the startingPort parameter. The two parameters are mutually exclusive.

### Scenario: Incrementing default port numbers from a starting point

The **wasprofile** command can assign port numbers based on a starting port value that you give on the command line, using the `-startingPort` parameter. The tool assigns port numbers sequentially from the starting port number value.

The order of port assignments is arbitrary. Predicting assignments is not possible.

For example, ports created with `-startingPort 20002` would appear similar to the following example:

#### Assigned ports for an Application Server profile

```

WC_defaulthost=20002
WC_adminhost=20003
WC_defaulthost_secure=20004
WC_adminhost_secure=20005
BOOTSTRAP_ADDRESS=20006
SOAP_CONNECTOR_ADDRESS=20007
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=20008
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=20009
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=20010
ORB_LISTENER_ADDRESS=20011
DCS_UNICAST_ADDRESS=20012
SIB_ENDPOINT_ADDRESS=20013
SIB_ENDPOINT_SECURE_ADDRESS=20014
SIB_MQ_ENDPOINT_ADDRESS=20015
SIB_MQ_ENDPOINT_SECURE_ADDRESS=20016

```

#### Assigned ports for a custom profile

```

WC_defaulthost=20002
WC_adminhost=20003
WC_defaulthost_secure=20004

```

```

WC_adminhost_secure=20005
BOOTSTRAP_ADDRESS=20006
SOAP_CONNECTOR_ADDRESS=20007
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=20008
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=20009
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=20010
ORB_LISTENER_ADDRESS=20011
CELL_DISCOVERY_ADDRESS=20012
DCS_UNICAST_ADDRESS=20013

```

### Assigned ports for a deployment manager profile

```

CELL_DISCOVERY_ADDRESS=20010
BOOTSTRAP_ADDRESS=20004
DRS_CLIENT_ADDRESS=7989
SOAP_CONNECTOR_ADDRESS=20005
ORB_LISTENER_ADDRESS=20009
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=20006
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=20008
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=20007
WC_adminhost=20002
DCS_UNICAST_ADDRESS=20011
WC_adminhost_secure=20003

```

### Example of startingPort parameter use

The following example of using the **wasprofile** command creates ports from an initial value of 20002, with the content shown in the previous example:

```

wasprofile.bat -create
               -profileName shasti
               -profilePath G:\shasti\WebSphere
               -templatePath template_path
               -nodeName W2K03
               -cellName W2K03_Cell01
               -hostName plane1nt
               -startingPort 20002

```

### Scenario: Setting up and using the profile environment

Most tasks that you perform in a profile are done using the **-profileName** attribute on the command line tools that you use. Such a scenario might be:

1. Create the server process using the *install\_root/bin/wasprofile.sh* (or *wasprofile.bat*) script from the original installation. Assume that you create the Profile02 profile.
2. In that command window or in another, change directories to the */bin* directory of the new server process.
3. Establish a temporary override for the normal WebSphere Application Server environment by using the **-profileName** attribute on a command you issue. In the same window, start *server1* by changing directories to the *install\_root/bin* directory of the original installation and issuing the command. There is no such command in the */bin* directory of the server process:

```
startServer.sh server1 -profileName Profile02
```

4. Notice the changes in the environment. Display all of the ports for the machine to see the ports that you set for the server process. For example, in a Linux bash shell or in the command window on a Windows platform, type:
5. Open a browser window and point it at the port defined for the HTTP\_TRANSPORT\_ADMIN port of the new process. For example, suppose the setting is HTTP\_TRANSPORT\_ADMIN=20003. Open the administrative console for *server1* by pointing your browser at:

```
http://hostname_orIP_address:20003/ibm/console/
```



## Scenario: Profile creation for a non-root user

Two methods exist for a non-root user to create a profile:

- The root user creates the profile and assigns ownership to the non-root user.
- A non-root user creates a profile after getting write permission to the appropriate directories.

**Remember:** An ease-of-use limitation exists for non-root users who create profiles. Mechanisms within the Profile creation wizard that suggest unique names and port values are disabled for non-root users. The non-root user must change the default field values in the Profile creation wizard for the profile name, node name, cell name, and port assignments. Consider assigning non-root users a range of values for each of the fields. You can assign responsibility to the non-root profilers for adhering to their proper value ranges and for maintaining the integrity of their own definitions.

**Root creates the profile and assigns ownership to a non-root user:** The root user can create a profile and assigns ownership of the profile directory to a non-root user.

1. The root user creates the profile with the following command:

```
./wasprofile.sh -create -profileName profile01 -profilePath install_root/profiles/profile01
```

2. The root user changes ownership of the directory for the profile to the non-root user with the following command:

```
chown -R user1 install_root/profiles/profile01
```

**A non-root user creates the profile (advanced option):** The root user can grant write permission to the appropriate files and directories to a non-root user. The non-root user can then create the profile. You can create a group for users who are authorized to create profiles. Or you can give everyone the ability to create profiles. The following example shows how to create a group that is authorized to create profiles.

1. Log on to the Application Server system as root.
2. Create the profilers group that you can use to create profiles.
3. Create a user named user1 to create profiles.
4. Add users root and user1 to the profilers group.
5. Log off and back on as root to pick up the new group.
6. As root, use operating system tools to change file permissions.

The following example assumes that the installation root directory is `/opt/IBM/WebSphere/AppServer`:

```
mkdir /opt/IBM/WebSphere/AppServer/logs/wasprofile
chgrp profilers /opt/IBM/WebSphere/AppServer/logs/wasprofile
chmod g+wr /opt/IBM/WebSphere/AppServer/logs/wasprofile
chgrp profilers /opt/IBM/WebSphere/AppServer/properties
chmod g+wr /opt/IBM/WebSphere/AppServer/properties
chmod g+wr /opt/IBM/WebSphere/AppServer/properties/profileRegistry.xml
```

You might have to change the permissions on additional `/opt/IBM/WebSphere/AppServer` directories if you encounter permission problems.

7. The non-root user who belongs to the profilers group can then create a profile in any directory to which the non-root user has write permission.

If the non-root user does not have write access to any directories, it is up to the root user to change that situation. If the non-root user does not have write access to the `/tmp` directory, it is up to the root user to change that as well.

The commands listed in step 6 give users assigned to the profilers group the ability to write to the `/opt/IBM/WebSphere/AppServer/logs/wasprofile` directory and to the `/opt/IBM/WebSphere/AppServer/properties` directory. It is not necessary to write to any other directories in the installation root of your WebSphere Application Server product.

Have non-root users create a `profiles` directory in their own area, not in the installation root directory of the product.

---

## Using the installation verification test

This topic describes how to use the installation verification test (IVT). The IVT verifies that the installation of the application server or deployment manager profile was successful. A *profile* consists of files that define the run-time environment for a deployment manager or an application server. Each profile has its own IVT command.

After installing the Network Deployment product and creating a deployment manager or application server profile, you are ready to use the installation verification test (IVT).

The IVT program scans product log files for errors and verifies core functionality of the product installation.

The Profile creation wizard creates profiles. After creating a profile, the Profile creation wizard displays a prompt for starting the First steps console. The First steps console is unique for each profile. See “firststeps command” on page 40 for more information.

Installation verification is the first option on the First steps console.



The IVT program for an application server profile starts and monitors the application server process, which is the server1 process. The installation verification for a deployment manager profile starts and monitors the deployment manager process, which is the dmgr process. The IVT works differently for the deployment manager profile than for a stand-alone application server. On a stand-alone application server, the IVT queries servlets from the ivtApp application. However, the deployment manager does not have the ivtApp application, so the IVT looks at log files only.

1. Start the First steps console and select **Installation verification** after creating a deployment manager profile or an application server profile.

No installation verification is possible for a custom profile. After federating the node and using the deployment manager to create a server, you can start the server process to verify its functionality.

Select the check box to launch the First steps console at the end of profile creation. You can also start the First steps console from the command line, as described in “firststeps command” on page 40.

You can also start the “ivt command” on page 101 directly from the bin directory of the profile:

-  `install_root/profiles/profile_name/bin/ivt.sh`
-  `install_root\profiles\profile_name\bin\ivt.bat`

If you create profiles in another location, the ivt script location is within the *profile\_home/bin* directory.

2. Observe the results in the First steps status window.

The default log file for installation verification is the *install\_root/profiles/profile\_name/logs/ivtClient.log*. If you create profiles in another location, the file path is *profile\_home/logs/ivtClient.log*.

The IVT provides the following useful information about the application server:

- The application server name
- The name of the profile
- The profile file path
- The type of profile
- The node name
- The current encoding
- The port number for the administrative console
- Various informational messages that include the location of the SystemOut.log file and how many errors are listed within the file
- A completion message

As the IVT starts the application server on a Windows platform, the IVT attempts to start the Windows service for the application server, if a Windows service exists. This is true even though the Windows service might have a manual startup type.

If you federate a stand-alone application server, you can still run the IVT on the server.

The IVT provides the following useful information about the deployment manager:

- The deployment manager server name: dmgr
- The name of the profile
- The profile file path
- The type of profile: dmgr
- The cell name
- The node name
- The current encoding
- The port number for the administrative console
- Various informational messages that include the location of the `SystemOut.log` file and how many errors are listed within the file
- A completion message

As the IVT starts the deployment manager on a Windows platform, the IVT attempts to start the Windows service for the deployment manager if a Windows service exists. This is true even though the Windows service might have a manual startup type.

See “Automatically restarting server processes” on page 235 for more information.

3. If the log shows that errors occurred during the installation verification, correct the errors and run the IVT again. If necessary, create a new profile after correcting the error, and run the IVT on the new profile.

The IVT program starts the server process automatically if the server is not running. Once the server initializes, the IVT runs a series of verification tests. The tool displays pass or fail status in a console window. The tool also logs results to the `profile_home/logs/ivtClient.log` file. As the IVT verifies your system, the tool reports any detectable errors in the `SystemOut.log` file.

Return to the *Installing your application serving environment* PDF to continue.

## ivt command

The **ivt** command starts the installation verification test (IVT) program. The IVT verifies that the installation of the application server or deployment manager profile was successful. A *profile* consists of files that define the run-time environment for a deployment manager or an application server. Each profile has its own IVT command.


The IVT program starts the application server or deployment manager automatically if the server process is not already running. After the server process initializes, the IVT runs a series of verification tests and displays pass or fail status in a console window.

The IVT program scans the `SystemOut.log` file for errors and verifies core functionality of the profile.

You can start the IVT program from the command line or from the First steps console.

### Location of the command file

The location of the `ivt.sh` or `ivt.bat` script for any profile is:

-  `install_root/profiles/profile_name/bin/ivt.sh`
-  `install_root\profiles\profile_name\bin\ivt.bat`

## Parameters

The following parameters are associated with this command.

### **server\_name**

Required parameter that identifies the name of the server process, such as server1 or dmgr.

### **profile\_name**

Required parameter that identifies the name of the profile that contains the server definition.

### **-p server\_port\_number**

Optional parameter that identifies the default\_host port when the port is not 9080, which is the default.

### **-host machine\_host\_name**

Optional parameter that identifies the host machine of the profile to test. The default is localhost.

## Syntax for the ivt command

Use the following syntax for the command:

- **UNIX** `install_root/profiles/profile_name/bin/ivt.sh`
- **Windows** `install_root\profiles\profile_name\bin\ivt.bat`

## Logging

The `ivt` command logs results to the `install_root/profiles/profile_name/logs/ivtClient.log` file.

## Example

The following examples test the server1 process in the profile01 profile on the myhost machine using the default\_host on port 9081.

### **Windows**

```
ivt.bat server1 profile01 -p 9081 -host myhost
```

### **UNIX**

```
ivt.sh server1 profile01 -p 9081 -host myhost
```

## Chapter 6. Configuring ports

This topic discusses configuring ports, particularly in coexistence scenarios.

1. Review “Port number settings in WebSphere Application Server versions.”

This topic provides reference information about identifying port numbers in versions of WebSphere Application Server, as a means of determining port conflicts that might occur when you intend for an earlier version to coexist with Version 6.

2. You can change port settings on the port assignment panel while using the Profile creation wizard.

See “Using the Profile creation wizard” on page 45 for more information.

3. After installation, edit the

`profiles_install_root/profile_name/config/cells/cell_name/nodes/node_name/serverindex.xml` file to change the port settings, or use scripting to change the values.

See the *Administering applications and their environment* PDF for more information.

### Port number settings in WebSphere Application Server versions

This topic provides reference information about identifying port numbers in versions of WebSphere Application Server, as a means of determining port conflicts that might occur when you intend for an earlier version to coexist or interoperate with Version 6.

#### Version 6 port numbers

Table 8. Port definitions for WebSphere Application Server Version 6

Port name	WebSphere Application Server	Network Deployment	File
	Value		
HTTP_TRANSPORT	9080	9080	
HTTP Admin Console Port (HTTP_TRANSPORT_ADMIN)	9060	9060	serverindex.xml and virtualhosts.xml
HTTPS Transport Port (HTTPS_TRANSPORT)	9443	9443	
HTTPS Admin Console Secure Port (HTTPS_TRANSPORT_ADMIN)	9043	9043	
BOOTSTRAP_ADDRESS	2809	9809	
SOAP_CONNECTOR-ADDRESS	8880	8879	
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	9401	9401	
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	9403	9403	
CSIV2_SSL_MULTIAUTH_LISTENER_ADDRESS	9402	9402	
ORB_LISTENER_ADDRESS	9100	9100	
DCS_UNICAST_ADDRESS	9353	9352	serverindex.xml
SIB_ENDPOINT_ADDRESS	7276	7276	
SIB_ENDPOINT_SECURE_ADDRESS	7286	7286	
SIB_MQ_ENDPOINT_ADDRESS	5558	5558	
SIB_MQ_ENDPOINT_SECURE_ADDRESS	5578	5578	
Internal JMS Server (JMSSERVER_SECURITY_PORT)	5557	Not applicable	
DRS_CLIENT_ADDRESS	7873	7989	

Table 8. Port definitions for WebSphere Application Server Version 6 (continued)

Port name	WebSphere Application Server	Network Deployment	File
	Value		
IBM HTTP Server Port	80	Not applicable	virtualhosts.xml, plugin-cfg.xml, and <i>IHSinstall_root/conf/</i> httpd.conf
IBM HTTPS Server Admin Port	8008	Not applicable	<i>IHSinstall_root/conf/</i> admin.conf
CELL_DISCOVERY_ADDRESS	Not applicable	7277	
CELL_MULTICAST_DISCOVERY_ADDRESS	Not applicable	7272	serverindex.xml
NODE_MULTICAST_IPV6_DISCOVERY_ADDRESS	5001	5001	

When you federate an Application Server node into a deployment manager cell, the deployment manager instantiates the nodeagent server process on the Application Server node. The nodeagent server uses these port assignments by default:

Table 9. Port definitions for the V6 nodeagent server process

Port name	Value	File
BOOTSTRAP_ADDRESS	2809	
ORB_LISTENER_ADDRESS	9100	
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	9901	
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	9202	
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	9201	
NODE_DISCOVERY_ADDRESS	7272	serverindex.xml
NODE_MULTICAST_DISCOVERY_ADDRESS	5000	
NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS	5001	
DCS_UNICAST_ADDRESS	9353	
DRS_CLIENT_ADDRESS	7888	
SOAP_CONNECTOR_ADDRESS	8878	

### Version 5.x port numbers

Table 10. Port definitions for WebSphere Application Server Version 5.x

Port name	WebSphere Application Server	Network Deployment	File
	Value		
HTTP_TRANSPORT	9080	Not applicable	
HTTPS Transport Port (HTTPS_TRANSPORT)	9443	Not applicable	
HTTP Admin Console Port (HTTP_TRANSPORT_ADMIN)	9090	9090	server.xml and virtualhosts.xml
HTTPS Admin Console Secure Port (HTTPS_TRANSPORT_ADMIN)	9043	9043	
Internal JMS Server (JMSSERVER_SECURITY_PORT)	5557	Not applicable	server.xml

Table 10. Port definitions for WebSphere Application Server Version 5.x (continued)

Port name	WebSphere Application Server	Network Deployment	File
	Value		
JMSSERVER_QUEUED_ADDRESS	5558	Not applicable	serverindex.xml
JMSSERVER_DIRECT_ADDRESS	5559	Not applicable	
BOOTSTRAP_ADDRESS	2809	9809	serverindex.xml
SOAP_CONNECTOR-ADDRESS	8880	8879	serverindex.xml
DRS_CLIENT_ADDRESS	7873	7989	serverindex.xml
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	0	9401	serverindex.xml
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	0	9403	serverindex.xml
CSIV2_SSL_MULTIAUTH_LISTENER_ADDRESS	0	9402	
IBM HTTP Server Port	80	Not applicable	virtualhosts.xml, plugin-cfg.xml, and <i>IHSinstall_root/conf/httpd.conf</i>
IBM HTTPS Server Admin Port	8008	Not applicable	<i>IHSinstall_root/conf/admin.conf</i>
CELL_DISCOVERY_ADDRESS	Not applicable	7277	
ORB_LISTENER_ADDRESS	9100	9100	serverindex.xml
CELL_MULTICAST_DISCOVERY_ADDRESS	Not applicable	7272	

When you federate an Application Server node into a deployment manager cell, the deployment manager instantiates the nodeagent server process on the Application Server node. The nodeagent server uses these port assignments by default:

Table 11. Port definitions for the V5.x nodeagent server process

Port name	Value	File
BOOTSTRAP_ADDRESS	2809	
ORB_LISTENER_ADDRESS	9900	
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	9901	
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	9101	
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	9201	serverindex.xml
NODE_DISCOVERY_ADDRESS	7272	
NODE_MULTICAST_DISCOVERY_ADDRESS	5000	
DRS_CLIENT_ADDRESS	7888	
SOAP_CONNECTOR_ADDRESS	8878	

### Version 4.0.x port numbers

**For WebSphere Application Server Advanced Single Server Edition, Version 4.0.x:** Inspect the `server-cfg.xml` file to find the Web container HTTP transports port values for the configuration.

**For WebSphere Application Server Advanced Edition, Version 4.0.x:** When the administrative server is running, use this command to extract the configuration from the database:

```
xmlConfig -export config.xml -nodeName theNodeName
```



Look for the Web container HTTP transports port assignments.

Table 12. Port definitions for WebSphere Application Server V4.0.x

Port name	Value	Advanced Edition	IBM WebSphere Business Integration Server Foundation Edition	Advanced Single Server Edition
			File	
bootstrapPort	900			
lsdPort	9000	admin.config	admin.config	
LSDSSLPort	9001			
HTTP transport port	9080			
HTTPS transport port	9443			server-cfg.xml
Admin Console HTTP transport port	9090	database	database	
ObjectLevelTrace	2102			
diagThreadPort	7000			

---

## Chapter 7. Communicating with Web servers

The WebSphere Application Server works with a Web server to route requests for dynamic content, such as servlets, from Web applications. The Web servers are necessary for directing traffic from browsers to the applications that run in WebSphere Application Server. The Web server plug-in uses the XML configuration file to determine whether a request is from the Web server or the Application Server.

The Web server plug-ins for distributed platform Web servers are provided on a separate CD from the WebSphere Application Server products. A Web Server Plug-in Installation Wizard is also provided on that CD. The *Installing your application serving environment* PDF describes how to install a Web server plug-in and create a Web server definition.

1. Install your Web server if it is not already installed. See the installation information provided with your Web server.
2. Ensure that your Web server is configured to perform operations required by Web applications, such as GET and POST. Typically, this involves setting a directive in the Web server configuration file (such as the `httpd.conf` file for an IBM HTTP Server). Refer to the Web server documentation for instructions. If an operation is not enabled when a servlet or JSP file requiring the operation is accessed, an error message displays, such as this one from the IBM HTTP Server:

HTTP method POST is not supported by this URL.

3. Use the Plug-in Installation wizard to install the appropriate plug-in file to your Web server and run the script `configureWeb_server_name` created by the wizard to create and configure the Web server definition in the WebSphere configuration repository. The following substeps are performed during the plug-in installation process. See the Plug-in Installation Roadmap for additional information.
  - a. A node is created. An unmanaged node is created when the Web server is on a different computer from the Application Server. An unmanaged node is a node that does not have a WebSphere node agent running on it. Using unmanaged nodes, WebSphere Application Server can represent servers that are not application servers within its configuration topology. This representation enables connection information between those servers and application servers to be maintained. “Managing nodes” on page 139 describes how to create a node.
  - b. A Web server definitions is created. You can also use either use the administrative console or use the `ConfigureWebServerDefintion.jacl` script to create a Web server definition.

If you use the administrative console:

    - 1) Select the node that was created in the preceding step, and in the Server name field, enter the local name of the Web server for which you are creating a Web server definition.
    - 2) Use the wizard to complete the Web server definition.
  - c. An application or modules are mapped to a Web server. If an application that you want to use with this Web server is already installed, the application is automatically mapped to the Web server. If the application is not installed, select this Web server during the Map modules to servers step of the application installation process.
  - d. Master repository is updated and saved.
4. **Optional:** Configure the plug-in. Use either the administrative console, or issue the `GenPluginCfg` command to create your `plugin-cfg.xml` file.

When setting up your Web server plug-in, you must decide whether or not to have the configuration automatically generated in response to a configuration change. When the Web server plug-in configuration service is enabled and any of the following conditions occur, the plug-in configuration file is automatically generated:

- When the Web server is created or saved.
- When an application is installed.
- When an application is uninstalled.
- When the virtual host definition is updated

Generating or regenerating the configuration file might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the Web server can access. If the Application Server is on the same physical machine as the Web server, the regeneration usually takes about 30 to 60 seconds to complete. The regeneration takes longer if they are not both on the same machine.

**Important:** When the plug-in configuration file is first generated, it does not include `admin_host` on the list of virtual hosts. The *Installing your application serving environment* PDF describes how to add it to the list.

To use the administrative console:

- a. Select **Servers > Web Servers > *webserver* > plug-in properties**.
- b. Select **Automatically generate plug-in configuration file** or click on one or more of the following topics to manually configure the `plugin-cfg.xml` file:
  - Caching
  - Request and response
  - Request routing
  - Service

Web server plug-in configuration properties maps each property to one of these topics.

- c. Click **OK**.
  - d. You might need to stop the application server and then start the application server again to enable the Web server to locate the `plugin-cfg.xml` file.
5. If you want to use Secure-socket layer (SSL) with this configuration, use the plug-in's installation wizard to install the appropriate GSKIT installation image file on your workstation. See the *Securing applications and their environment* PDF for information on how to configure GSKIT.
  6. If you want to enable the Web server plug-in to use private headers, define an SSL configuration repertoire that defines a trust file. Then in the administrative console, select **Application servers > server1 > Web Container Settings > Web Container Transport Chains > *transport\_chain* > SSL Inbound Channel (SSL\_2)** and specify this repertoire for that transport chain. If you try to use private headers without setting up an SSL configuration repertoire that does not include a trust file definition, the private headers will be ignored. If the private headers are ignored, the application server might not locate the requested application.

After you enable the use of private headers, the transport chain's SSL inbound channel trusts all private headers it receives. Therefore, you must ensure that all paths to the transport chain's SSL inbound channel are trusted.

7. Tune your Web server with Web server tuning parameters.
8. Propagate the plug-in configuration. The plug-in configuration file (`plugin-cfg.xml`) is automatically propagated to the Web server if the Web server plug-in configuration service is enabled, and one of the following is true:
  - The Web server is a local Web server. (It are located on the same machine as an application server.)
  - The Web server is a remote IBM HTTP Server (IHS) Version 6.0 that has a running IHS Administrative server.

If neither of these conditions is true, the `plugin-cfg.xml` file must be manually copied to the remote Web server's installation location.

The remote Web server installation location is the location you specified when you created the node for this Web server.

The configuration is complete. To activate the configuration, stop and restart the Web server. If you encounter problems restarting your Web server, check the `http_plugin.log` file for information on what portion of the `plugin-cfg.xml` file contains an error. The log file states the line number on which the error occurred along with other details that might help you diagnose why the Web server did not start. You can then use the administrative console to update the `plugin-cfg.xml` file.

If applications are infrequently installed or uninstalled, which is usually the situation in a production environment, or if you can tolerate the performance impact of generating and distributing the plug-in configuration file each time any of the previously listed actions occur, you should consider enabling this service.

If you are making a series of simultaneous changes, like installing numerous applications, you might want the configuration service disabled until after you make the last change. The Web server plug-in configuration service is enabled by default. To disable this service, in the administrative console click **Servers > Application Servers > *server\_name* > Administration Services > Web server plug-in configuration service** and then unselect the Enable automated Web server configuration processing option.

**Tip:** If your installation uses a firewall, make sure you configure the Web server plug-in to use a port that has been opened. (See your security administrator for information on how to obtain an open port.)

---

## Web server plug-in properties settings

Use this page to view or change the settings of a Web server plug-in configuration file. The plug-in configuration file, `plugin_cfg.xml`, provides properties for establishing communication between the Web server and the Application Server.

To view this administrative console page, click **Servers > Web Servers > *Web\_server\_name* Plug-in Properties**.

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 this Web server has accessed applications running on application servers and there is an `http_plugin.log` file.

### Plug-in log file name

Specifies the fully qualified path to the log file to which the plug-in will write error messages. The default file path is `plugin_install_root/logs/web_server_name/http_plugin.log`.

If the file does not exist then it will be created. If the file already exists, it will be opened in append mode and the previous plug-in log messages will remain.

This field corresponds to the `RequestMetrics loggingEnabled` element in the `plugin-cfg.xml` file.

<b>Data type</b>	String
<b>Default for Linux and UNIX platforms</b>	<code>plugin_install_root/logs/web_server_name/http_plugin.log</code>
<b>Default for Windows platforms</b>	<code>plugin_install_root/logs/web_server_name/http_plugin.log</code>

### Plug-in installation location

Specifies the fully qualified path to where the plug-in configuration file is installed.

<b>Data type</b>	String
<b>Default</b>	The default value is the installation root directory.

If you select a Web server plug-in during installation, the installer program configures the Web server to identify the location of the `plugin-cfg.xml` file, if possible. The Web server is considered installed on a local machine if it is on the same machine as the application server. It is considered installed on a remote machine if the Web server and the application server are on different machines.

- If the Web server is installed on a remote machine, the plug-in configuration file, by default, will be installed in the `plugin_install_root/config/web_server_name` directory.
- If the Web server is installed on a local standalone machine, the plug-in configuration file, by default will be installed in the `profile_install_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name` directory.
- If the Web server is installed on a local distributed machine, the plug-in configuration file, by default will be installed in the `profile_install_root/config/cells/cell_name/nodes/node_name/servers/web_server_name` directory.

The installer program adds a directive to the Web server configuration that specifies the location of the `plugin-cfg.xml` file.

For remote Web servers, you must copy the file from the local directory where the Application Server is installed to the remote machine. This is known as propagating the plug-in configuration file. If you are using an IBM HTTP Server (IHS) V6 for your Web server, WebSphere Application Server can automatically propagate the plug-in configuration file for you to remote machines provided there is a working HTTP transport mechanism to propagate the file.

## Plug-in configuration file name

Specifies the file name of the configuration file for the plug-in. The Application Server generates the `plugin-cfg.xml` file by default. The configuration file identifies applications, Application Servers, clusters, and HTTP ports for the Web server. The Web server uses the file to access deployed applications on various Application Servers.

<b>Data type</b>	String
<b>Default</b>	<code>plugin-cfg.xml</code>

If you select a plug-in during installation, the installer program configures the Web server to identify the location and name of the `plugin-cfg.xml` file, if possible.

You can change the name of the plug-in configuration file. However, if you do change the file name, you must also change the Web server configuration to point to the new plug-in configuration file.

## Automatically generate plug-in configuration file

To automatically generate a plug-in configuration file to a remote Web server:

- This field must be checked.
- The plug-in configuration service must be enabled

When the plug-in configuration service is enabled, a plug-in configuration file is automatically generated for a Web server whenever:

- The WebSphere Application Server administrator defines new Web server.
- An application is deployed to an Application Server.
- An application is uninstalled.
- A virtual host definition is updated and saved.

Clear the check box if you want to manually generate a plug-in configuration file for this Web server.

**Important:** When the plug-in configuration file is generated, it does not include `admin_host` on the list of virtual hosts. The *Installing your application serving environment* PDF describes how to add it to the list.

## Automatically propagate plug-in configuration file

To automatically propagate a copy of a changed plug-in configuration file to a Web server:

- This field must be checked.
- The plug-in configuration service must be enabled
- A WebSphere Application Server node agent must be on the node that hosts the Web server associated with the changed plug-in configuration file.

**Note:** The plug-in configuration file can only be automatically propagated to a remote Web server if that Web server is an IHS V6.0 Web server and its administration server is running.

## Ignore DNS failures during Web server startup

Specifies whether the plug-in ignores DNS failures within a configuration when starting.

This field corresponds to the `IgnoreDNSFailures` element in the `plugin-cfg.xml` file.

When set to **true**, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each `ServerCluster` is able to resolve the host name. Any server for which the host name can not be resolved is marked **unavailable** for the life of the configuration. No attempts to resolve the host name are made later on during the routing of requests. If a DNS failure occurs, a log message is written to the plug-in log file and the plug-in initialization continues rather than causing the Web server not to start. When **false** is specified, DNS failures cause the Web server not to start.

<b>Data type</b>	String
<b>Default</b>	false

## Refresh configuration interval

Specifies the time interval, in seconds, at which the plug-in should check the configuration file to see if updates or changes have occurred. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 seconds is preferable. In production, a higher value than the default is preferable because updates to the configuration will not occur so often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in configuration file successfully reloads. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

<b>Data type</b>	Integer
<b>Default</b>	60 seconds.

## Plug-in logging

Specifies the location and name of the `http_plugin.log` file. Also specifies the scope of messages in the log.

This field corresponds to the `RequestMetrics traceLevel` element in the `plugin-cfg.xml` file.

The log describes the location and level of log messages that are written by the plug-in. If a log is not specified within the configuration file, then, in some cases, log messages are written to the Web server error log.

On a distributed platform, if the log file does not exist then it will be created. If the log file already exists, it will be opened in append mode and the previous plug-in log messages will remain.

**Log file name** - The fully qualified path to the log file to which the plug-in will write error messages.

<b>Data type</b>	String
<b>Default</b>	<code>plugin_install_root/logs/web_server_name/http_plugin.log</code>

Specify the file path of the `http_plugin.log` file.

**Log level**- The level of detail of the log messages that the plug-in should write to the log. You can specify one of the following values for this attribute:

- Trace. All of the steps in the request process are logged in detail.
- Stats. The server selected for each request and other load balancing information relating to request handling is logged.
- Warn. All warning and error messages resulting from abnormal request processing are logged.
- Error. Only error messages resulting from abnormal request processing are logged.

If a Log level is not specified, the default value **Error** is used.

Be careful when setting the level to **Trace**. A lot of messages are logged at this level which can cause the disk space/file system to fill up very quickly. A **Trace** setting should never be used in a normally functioning environment as it adversely affects performance.

<b>Data type</b>	String
<b>Default</b>	Error

## Web server plug-in request and response optimization properties settings

Use this page to view or change the request and response optimization properties for a Web server plug-in.

To view this administrative console page, click **Servers > Web Servers > web\_server\_name Plug-in Properties > Request and Response**.

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 this Web server has accessed applications running on application servers and there is an `http_plugin.log` file.

### Maximum chunk size used when reading the response body

Specifies the maximum chunk size the plug-in can use when reading the response body.

This field corresponds to the `ResponseChunkSize` element in the `plugin-cfg.xml` file.

The plug-in reads the response body in 64K chunks until all of the response data is read. This approach causes a performance problem for requests whose response body contains large amounts of data.

If the content length of the response body is unknown, the values specified for this property is used as the size of the buffer that is allocated. The response body is then read in this size chunks, until the entire body is read. If the content length is known, then a buffer size of either the content length or the specified size (whichever is less) is used to read the response body.

<b>Data type</b>	Integer
<b>Default</b>	64 kilobytes

Specify the size in kilobytes (1024 byte blocks).



## **Enable Nagle algorithm for connections to the Application Server**

When checked, the Nagle algorithm is enabled for connections between the plug-in and the Application Server.

This field corresponds to the ASDisableNagle element in the plugin-cfg.xml file.

The Nagle algorithm is named after engineer John Nagle, who invented this standard part of the transmission control protocol/internet protocol (TCP/IP). The algorithm reduces network overhead by adding a transmission delay (usually 20 milliseconds) to a small packet, which lets other small packets arrive and be included in the transmission. Because communications has an associated cost that is not as dependent on packet size as it is on frequency of transmission, this algorithm potentially reduces overhead with a more efficient number of transmissions.

Clear the check box to disable the Nagle algorithm.

## **Enable Nagle Algorithm for the IIS Web Server**

When checked, the Nagle algorithm is used for connections from the Microsoft Internet Informations Services (IIS) Web Server to the Application Server.

This field corresponds to the IHSDisableNagle element in the plugin-cfg.xml file. It only appears if you are using the Microsoft Internet Informations Services (IIS) Web server. Clear the check box to disable the Nagle algorithm for this connection.

## **Chunk response to the client**

When checked, responses to the client are broken into chunks if a Transfer-Encoding : Chunked response header is present in the response.

This field corresponds to the ChunkedResponse element in the plugin-cfg.xml file. It only appears if you are using a Microsoft Internet Informations Services (IIS) Web Server, a Java System Web server, or a Domino Web server. The IIS Web server automatically handles breaking the response into chunks to send to the client.

Clear the check box to if you do not want responses broken into chunks.

## **Accept content for all requests**

This field corresponds to the AcceptAllContent element in the plugin-cfg.xml file.

When selected, users can include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

## **Virtual host matching**

When selected, virtual host mapping is performed by physically using the port number for which the request was received.

This field corresponds to the VHostMatchingCompat element in the plugin-cfg.xml file.

If this field is not selected, matching is done logically using the port number contained in the host header.

Use the radio buttons to make your physical or logical port selection.

## **Application server port preference**

Specifies which port number the Application Server should use to build URI's for a sendRedirect.

This field corresponds to the AppServerPortPreference element in the plugin-cfg.xml file.

Specify:

- `webserverPort` if the port number from the host header of the HTTP request coming in is to be used.
- `hostHeader` if the port number on which the Web server received the request is to be used.

The default is `webserverPort`.

### **Priority used by the IIS Web server when loading the plug-in configuration file**

Specifies the priority in which the Microsoft Internet Informations Services (IIS) Web server loads the WebSphere Web server plug-in.

This field corresponds to the `IISPluginPriority` element in the `plugin-cfg.xml` file. It only appears if you are using the IIS Web server. Because the IIS Web server uses this value during startup, the Web server must be restarted before a change to this field takes effect.

Select one of the following priorities:

- High
- Medium
- Low

The default value of **High** ensures that all requests are handled by the Web server plug-in before they are handled by any other filter/extensions. If problems occur while using a priority of **Medium** or **Low**, you will have to rearrange the order or change the priority of the interfering filter/extension.

### **Web server plug-in caching properties settings**

Use this page to view or change the caching properties for a Web server plug-in.

To view this administrative console page, click **Servers > Web Servers > *Web\_server\_name* Plug-in Properties > Caching Properties**.

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 this Web server has accessed applications running on application servers and there is an `http_plugin.log` file.

### **Enable Edge Side Include (ESI) processing to cache the responses**

Specifies whether to enable Edge Side Include processing to cache the responses.

This field corresponds to the `esiEnable` element in the `plugin-cfg.xml` file.

When selected, Edge Side Include (ESI) processing is used to cache responses. If ESI processing is disabled for the plug-in, the other ESI plug-in properties are ignored. Clear the checkbox to disable Edge Side Include processing.

### **Enable invalidation monitor to receive notifications**

When checked, the ESI processor receives invalidations from the Application Server.

This field corresponds to the `ESIInvalidationMonitor` element in the `plugin-cfg.xml` file. It is ignored if Edge Side Include (ESI) processing is not enabled for the plug-in.

### **Maximum cache size**

Specifies, in 1K byte units, the maximum size of the cache. The default maximum size of the cache is 1024K bytes (1 megabyte). If the cache is full, the first entry to be evicted from the cache is the entry that is closest its expiration time.

This field corresponds to the esiMaxCacheSize element in the plugin-cfg.xml file.

<b>Data type</b>	Integer
<b>Default</b>	1024 kilobytes

Specify the size in kilobytes (1024 byte blocks).

## Web server plug-in request routing properties settings

Use this page to view or change the request routing properties for a Web server plug-in.

To view this administrative console page, click **Servers > Web Servers > *Web\_server\_name* Plug-in Properties > Plug-in *server\_cluster\_name* Properties**.

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 this Web server has accessed applications running on application servers and there is an http\_plugin.log file.

### Load balancing option

Specifies the load balancing option that the plug-in uses in sending requests to the various application servers associated with that Web server.

This field corresponds to the LoadBalanceWeight element in the plugin-cfg.xml file.

Select the appropriate load balancing option:

- Round robin
- Random

The Round Robin implementation has a random starting point. The first application server is picked randomly. Round Robin is then used to pick application servers from that point forward. This implementation ensures that in multiple process based Web servers, all of the processes don't start up by sending the first request to the same Application Server.

The default load balancing type is Round Robin.

### Retry interval

Specifies the length of time, in seconds, that should elapse from the time an application server is marked down to the time that the plug-in retries a connection.

This field corresponds to the ServerWaitforContinue element in the plugin-cfg.xml file.

<b>Data type</b>	Integer
<b>Default</b>	60 seconds

### Maximum size of request content

Select whether there is a limit on the size of request content. If limited, this field also specifies the maximum number of bytes of request content allowed in order for the plug-in to attempt to send the request to an application server.

This field corresponds to the PostSizeLimit element in the plugin-cfg.xml file.

When a limit is set, the plug-in fails any request that is received that is greater than the specified limit.

Select whether to limit the size of request content:

- No limit
- Set limit

If **Set limit** is selected, specify a limit size.

<b>Data type</b>	Integer
<b>Default</b>	-1, which indicates there is no limit for the post size.  Specify the size in kilobytes (1024 byte blocks).

## Remove special headers

When checked, the plug-in will remove any headers from incoming requests before adding the headers the plug-in is supposed to add before forwarding the request to an application server.

This field corresponds to the `RemoveSpecialHeaders` element in the `plugin-cfg.xml` file.

The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. Not removing the headers from incoming requests introduces a potential security exposure.

Clear the check box to retain special headers.

## Clone separator change

When checked, the plug-in expects the plus character (+) as the clone separator.

This field corresponds to the `ServerCloneID` element in the `plugin-cfg.xml` file.

Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. If this field is checked, you must also change the configurations of the associated application servers such that the application servers separates clone IDs with the plus character as well.

Clear the checkbox to use the colon character to separate clone IDs.

---

## Web server plug-in custom properties

If you are using a Web server plug-in, you can add the following custom property to the configuration settings for that plug-in.

To add a custom property:

1. In the administrative console, click **Servers > Web Servers > *Web\_server\_name* > Plug-in properties > Custom properties > New**
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 is a list of Web server plug-in custom properties that are provided with the Application Server. These properties are not shown on the properties settings pages for the plug-in.

## StashfileLocation

Use this element to set a value for the stashfile initialization parameter.

**Data type** String

## KeyringLocation

Use this element to set a value for the keyring initialization parameter.

**Data type** String

---

## Web server plug-in configuration service properties settings

Use this page to view or change the configuration settings for the Web server plug-in configuration service.

To view this administrative console page, click **Application Servers** > *server\_name* > **Administration Services** > **Web server plug-in configuration service**.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

## Enable automated Web server configuration processing

When selected, the Web server plug-in configuration service automatically generates the plug-in configuration file whenever the Web server environment changes. For example, the plug-in configuration file is regenerated whenever one of the following activities occurs:

- A new application is deployed on an associated application server.
- The Web server definition is saved.
- An application is removed from an associated application server.
- A new virtual host is defined.

---

## Application Server property settings for a Web server plug-in

Use this page to view or change application server settings for a Web server plug-in.

To view this administrative console page, click **Application Servers** > *server\_name* > **Web Server Plug-in**.

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 this Web server has accessed applications running on application servers and there is an `http_plugin.log` file.

## Server role

Specifies the role this application server is assigned.

Select **Primary** to add this application server to the list of primary application servers. The plug-in initially attempts to route requests to the application servers on this list.

Select **Backup** to add this application server to the list of backup application servers. The plug-in does not load balance across the backup application servers. A backup server is only used if a primary server is not available. When the plug-in determines that a backup application server is required, it goes through the list of backup servers, in order, until no servers are left in the list or until a request is successfully sent and a response received from one of the servers on this list.

## Connect timeout

Specifies whether or not there is a limited amount of time the Application Server will maintain a connection with the Web server.

You can select either **No timeout** or **Set timeout**. If you select **Set timeout** you, must specify, in seconds, the length of time a connection with the Web server is to be maintained.

This property enables the plug-in to perform non-blocking connections with the application server. Non-blocking connections are beneficial when the plug-in is unable to contact the destination to determine whether or not the port is available. If no value is specified for this property, the plug-in performs a blocking connect in which the plug-in sits until an operating system times out (which could be as long as 2 minutes depending on the platform) and allows the plug-in to mark the server unavailable.

A value of 0 causes the plug-in to perform a blocking connect. A value greater than 0 specifies the number of seconds you want the plug-in to wait for a successful connection. If a connection does not occur after that time interval, the plug-in marks the server unavailable and fails over to another application server defined for the requested application.

**Data type** Integer

## Maximum number of connections that can be handled by the Application Server

Specifies the maximum number of pending connections to an Application Server that can be flowing through a Web server process at any point in time.

This field corresponds to the ServerMaxConnections element in the plugin-cfg.xml file.

You can select either **No limit** or **Set limit**. If you select **Set limit** you, must specify the maximum number of connections that can exist between the Web server and the Application Server at any given point in time.

For example, assuming that:

- The application server is fronted by 5 nodes that are running an IHS Web server.
- Each node starts 2 processes.
- This property is set to 50.

In this example, the application server could potentially get up to 500 connections. (You take the number of nodes, 5, multiply it by the number of processes, 2, and then multiply that number by the number specified for this property, 50, for a total of 500 connections.)

If this attribute is set to either zero or -1, there is no limit to the number of pending connections to the Application Servers.

**Data type** Integer  
**Default** -1

## Use extended handshake to check whether Application Server is running

When selected, the Web server plug-in will use an extended handshake to check whether or not the Application Server is running.

This field corresponds to the ServerExtendedHandshake element in the plugin-cfg.xml file.

Select this property if a proxy firewall is between the plug-in and the application server.

The plug-in marks a server as down when the connect() fails. However, when a proxy firewall is in between the plug-in and the application server, the connect() will succeed, even though the back end application server is down. This causes the plug-in to not failover correctly to other application servers.

If the plug-in performs some handshaking with the application server to ensure that it is started before it sends a request it can failover to another application server if it detects that the application server with which it is attempting to perform a handshake is down.

## Send the header "100 Continue" before sending the request content

This field corresponds to the WaitForContinue element in the plugin-cfg.xml file.

When selected, the Web server plug-in will send the header "100 Continue" to the Application Server before it sends the request content.

---

## Web server plug-in configuration properties

The following table indicates which panel in the administrative console you need to use to manually configure a Web server plug-in property.

Table 13. Web server plug-in configuration properties

Administrative console panel	Field name	Configuration property name
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties</b>	Refresh configuration interval	RefreshInterval
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties</b>	Plug-in log file name	Log->name
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties</b>	Plug-in logging	Log->LogLevel
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties</b>	Ignore DNS failures during Web server startup	IgnoreDNSFailures
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Custom properties &gt; New</b>	KeyringLocation	Keyring
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Custom properties &gt; New</b>	StashfileLocation	Stashfile
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request routing</b>	Load balancing option	LoadBalance



Table 13. Web server plug-in configuration properties (continued)

In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request Routing</b>	Clone separator change	CloneSeparatorChange
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request Routing</b>	Retry interval	RetryInterval
In the administrative console, click <b>Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request routing</b>	Maximum size of request content	PostSizeLimit
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request routing</b>	Remove special headers	RemoveSpecialHeaders
In the administrative console, click <b>Application Servers &gt; server_name &gt; Web server plug-in properties</b>	Server role	PrimaryServers and BackupServers list
In the administrative console, click <b>Application Servers &gt; server_name &gt; Web server plug-in properties</b>	Connect timeout	Server ConnectTimeout
In the administrative console, click <b>Application Servers &gt; server_name &gt; Web server plug-in properties</b>	Use extended handshake to check whether Application Server is running	Server Extended Handshake
In the administrative console, click <b>Application Servers &gt; server_name &gt; Web server plug-in properties</b>	Send the header "100 Continue" before sending the request content	WaitForContinue
In the administrative console, click <b>Application Servers &gt; server_name &gt; Web server plug-in properties</b>	Maximum number of connections that can be handled by the Application Server	Server MaxConnections
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Application server port preference	AppServerPortPreference
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Enable Nagle algorithm for connections to the Application Server	ASDisableViewNagle
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Enable Nagle Algorithm for the IIS Web Server	IISDisableViewNagle
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Virtual host matching	VHostMatchingCompat

Table 13. Web server plug-in configuration properties (continued)

In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Maximum chunk size used when reading the response body	ResponseChunkSize
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Accept content for all requests	AcceptAllContent
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Chunk response to the client	ChunkedResponse
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Request and Response</b>	Priority used by the IIS Web server when loading the plug-in configuration file	IISPluginPriority
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Caching</b>	Enable Edge Side Include (ESI) processing to cache the responses	ESIEnable
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Caching</b>	Maximum cache size	ESIMaxCacheSize
In the administrative console, click <b>Servers &gt; Web Servers &gt; Web_server_name &gt; Plug-in properties &gt; Caching</b>	Enable invalidation monitor to receive notifications	ESIInvalidationMonitor

## Web server plug-in connections

The WebSphere Application Server Web server plug-ins are used to establish and maintain persistent HTTP and HTTPS connections to Application Servers .

When the plug-in is ready to send a request to the application server, it first checks its connection pool for existing connections. If an existing connection is available the plug-in checks its connection status. If the status is still good, the plug-in uses that connection to send the request. If a connection does not exist, the plug-in creates one. If a connection exists but has been closed by the application server, the plug-in closes that connection and opens a new one.

After a connection is established between a plug-in and an application server, it will not be closed unless the application server closes it for one of the following reasons:

- If the **Use Keep-Alive** property is selected and the time limit specified on the **Read timeout** or **Write timeout** property for the HTTP inbound channel has expired.
- The maximum number of persistent requests which can be processed on an HTTP inbound channel has been exceeded. (This number is set using the HTTP inbound channel's **Maximum persistent requests** property.)
- The Application Server is shutting down.

Even if the application server closes a connection, the plug-in will not know that it has been closed until it tries to use it again. The connection will be closed if one of the following events occur:

- The plug-in receives a new HTTP request and tries to reuse the existing connection.
- The number of **httpd** processes drop because the Web server is not receiving any new HTTP requests. (For the IHS Web server, the number of **httpd** processes that are kept alive depends on the value specified on the Web server's **MinSpareServers** directive.)
- The Web server is stopped and all **httpd** processes are terminated, and their corresponding sockets are closed.

**Note:** Sometimes, if a heavy request load is stopped or decreased abruptly on a particular application server, a lot of the plug-in's connections to that application server will be in **CLOSE\_WAIT** state. Because these connections will be closed the first time the plug-in tries to reuse them, having a large number of connections in **CLOSE\_WAIT** state should not affect performance

---

## Web server plug-in remote user information processing

You can configure your Web server with a third-party authentication module and then configure the Web server plug-in to route requests to the Application Server. If an application calls the `getRemoteUser()` method, it relies on a private HTTP header that contains the remote user information and is parsed by the plug-in. The plug-in sets the private HTTP header value whenever a Web server authentication module populates the remote user in the Web server data structure. If the private HTTP header value is not set, the application's call to `getRemoteUser()` returns a null value.

- In the case of an Apache and IBM HTTP Server (IHS) Web server, the plug-in builds the private header from the information contained in the associated request record.
- In the case of a Sun One Web server, the plug-in builds the private header from the information contained in the **auth\_user** property associated with the request. The private header is usually set to the name of the local HTTP user of the Web browser, if HTTP access authorization is activated for the URL.
- In the case of a Domino Web server, the plug-in builds the private header from the information contained in the **REMOTE\_USER** environment variable. The plug-in sets this variable to **anonymous** for users who have not logged in and to the *username* for users who are logged into the application.
- In the case of an Internet Information Services (IIS) Web server, the plug-in builds the private header from the information contained in the **REMOTE\_USER** environment variable. The plug-in sets this variable to the name of the user as it is derived from the authorization header sent by the client.

If the private header is not being set in the Sun One, IIS, or Domino Web server plug-in, make sure the request record includes information about the user requesting the data.

**Note:** If an application's call to `getRemoteUser()` returns a null value, or if the correct remote user information is not being added to the Web server plug-in's data structure, make sure the remote user parameter within the `WebAgent` is still set to **YES**. (Sometimes this parameter gets set to **NO** when service is applied.)

---

## Web server plug-ins

Web server plug-ins enable the Web server to communicate requests for dynamic content, such as servlets, to the application server. A Web server plug-in is associated with each Web server definition. The configuration file (`plugin-cfg.xml`) that is generated for each plug-in is based on the applications that are routed through the associated Web server.

A Web server plug-in is used to forward HTTP requests from a supported Web server to an application server. Using a Web server plug-in to provide communication between a Web server and an application server has the following advantages:

- XML-based configuration file

- Standard protocol recognized by firewall products
- Security using HTTPS, replacing proprietary Open Servlet Engine (OSE) over Secure Sockets Layer (SSL)

Each of the supported distributed platform Web server plug-ins run on a number of operating systems. See Supported Hardware and Software at <http://www.ibm.com/software/webservers/appserv/doc/latest/prereq.html> for the product for the most current information about supported Web servers.

---

## Checking your IBM HTTP Server version

At times, you might need to determine the version of your IBM HTTP Server installation.

1. Change directory to the installation root of the Web server. For example, this is /opt/IBMIHS on a Solaris machine.
2. Find the subdirectory that contains apache.exe (on a Windows platforms) or apachectl (on a UNIX-based platforms, such as z/OS, Solaris, Linux, and HP-UX)
3. On a Windows platform, issue:  

```
apache.exe -V
```
4. On a UNIX-based platforms issue:  

```
./apachectl -V 4
```

The version is shown in the "Server version:" field and will look something like the following:

```
Server version: IBM_HTTP_Server/2.0.47 Apache/2.0.47
Server built: July 2 2004 20:38:36
Server's Module Magic Number: 20020903:4.
```

---

## Web server tuning parameters

WebSphere Application Server provides plug-ins for several Web server brands and versions. Each Web server operating system combination has specific tuning parameters that affect the application performance.

### • IBM HTTP Server

The IBM HTTP Server V6.0 is a multi-process, multi-threaded server.

#### – Access logs

- **Description:** Collects all incoming HTTP requests. Logging degrades performance because IO operation overhead causes logs to grow significantly in a short time.
- **How to view or set:**
  1. Open the IBM HTTP Server httpd.conf file, located in the directory *IBM\_HTTP\_Server\_root\_directory/conf*.
  2. Search for a line with the text **CustomLog**.
  3. Comment out this line by placing # in front of the line.
  4. Save and close the httpd.conf file.
  5. Stop and restart the IBM HTTP Server.
- **Default value:** Logging of every incoming HTTP request is enabled.
- **Recommended value:** Disable the access logs.

#### – MaxClients

- **Description:** The MaxClients directive controls the maximum number of simultaneous connections or users that the web server can service at any one time. If, at peak usage, your web server needs to support 200 active users at once, you should set MaxClients to 220 (200 plus an extra 10% for load growth). Setting MaxClients too low could cause some users to believe the web server is not responding. You should have sufficient RAM in your web server machines to support each connected client. For IBM HTTP Server V6.0 on UNIX, you should allocate around 1.5MB MaxClients of RAM for use by the IBM HTTP Server. For IBM HTTP Server V6.0 on Windows,

you should allocate around 300KB MaxClients of RAM for use by the IBM HTTP Server. Some third party modules can significantly increase the amount of RAM used per connected client.

- **How to view or set:** Edit the MaxClients directive in the IBM HTTP Server httpd.conf file, located in the directory *IBM\_HTTP\_Server\_root\_directory/conf*.
- **Default value:** 150
- **Recommended value:** The maximum number of users normally simultaneously connected to your web server, plus an additional 10% for buffer. Note: The KeepAliveTimeout setting can affect how long a user is connected to the webserver.
- **MinSpareServers, MaxSpareServers, and StartServers**
  - **Description:** Pre-allocates and maintains the specified number of processes so that few processes are created and destroyed as the load approaches the specified number of processes. Specifying similar values reduces the CPU usage for creating and destroying HTTPD processes. Adjust this parameter if the time waiting for IBM HTTP Server to start more servers, so that it can handle HTTP requests, is not acceptable.
  - **How to view or set:** Edit the MinSpareServers, MaxSpareServers and StartServers directives in the httpd.conf file located in the *IBM\_HTTP\_Server\_root\_directory/conf* directory.
  - **Default value:** MinSpareServers 5, MaxSpareServers 10, StartServers 5
  - **Recommended value:** For optimum performance, specify the same value for the MinSpareServers and the StartServers parameters. If MaxSpareServers is set to less than MinSpareServers, IBM HTTP Server resets MaxSpareServer=MinSpareServer+1. Setting the StartServers too high can cause swapping if memory is not sufficient, degrading performance.
- **ListenBackLog**
  - **Description:** Sets the length of a pending connections queue. When several clients request connections to the IBM HTTP Server, and all threads used, a queue exists to hold additional client requests. However, if you use the default Fast Response Cache Accelerator (FRCA) feature of IBM HTTP Server V6.0 on Windows, the ListenBackLog directive is not used since FRCA has its own internal queue.
  - **How to view or set:** For non-FRCA: Edit the IBM HTTP Server httpd.conf file. Then, add or view the ListenBackLog directive.
  - **Default value:** For HTTP Server V6.0: 1024 with FRCA enabled, 511 with FRCA disabled
  - **Recommended value:** Use the defaults.
- **IBM HTTP Server - Linux**
  - **MaxRequestsPerChild**
    - **Description:** Sets the limit on the number of requests that an individual child server process handles. After the number of requests reaches the value set for the MaxRequestsPerChild parameter, the child process dies. Adjust this parameter if destroying and creating child processes is degrading your Web server performance.
    - **How to view or set:**
      1. Edit the IBM HTTP server httpd.conf file located in the *IBM\_HTTP\_Server\_root\_directory/conf* directory.
      2. Change the value of the parameter.
      3. Save the changes and restart the IBM HTTP server.
    - **Default value:** 500
    - **Recommended value:** Should normally be set to 0. Non-zero settings can be useful if child memory usage is observed to steadily increase over time. Memory leaks have occasionally been observed in third party modules and various OS runtime libraries used by the IBM HTTP Server.
- **IBM HTTP Server - Windows 2000 and Windows 2003**
  - **ThreadsPerChild**
    - **Description:** Sets the number of concurrent threads running at any one time within the IBM HTTP Server.
    - **How to view or set:** Edit the IBM HTTP Server file httpd.conf file located in the directory *IBM\_HTTP\_Server\_root\_directory/conf*. Change the value of the parameter. Save the changes and restart the IBM HTTP Server.

There are two ways to find how many threads are used under load:

      1. Use the Windows 2000 and Windows 2003 Performance Monitor under the desktop Start menu:

- a. Right-click the status bar on your desktop. Click **Task Manager**.
- b. Select the **Processes** tab.
- c. Click **View > Select Columns**.
- d. Select **Thread Count**.
- e. Click **OK**.

The **Processes** tab shows the number threads for each process under the column name **Threads**, including Apache.

2. Use the IBM HTTP Server server-status (this choice works on all platforms, not just Windows):
  - a. Edit the IBM HTTP Server httpd.conf file as follows: Remove the comment character # from the following lines: #LoadModule status\_module, modules/ApacheModuleStatus.dll, #<Location/server-status>, #SetHandler server-status, and #</Location>.
  - b. Save the changes and restart the IBM HTTP Server.
  - c. In a Web browser, go to the URL: http://yourhost/server-status. Alternatively,
  - d. Click **Reload** to update status.
  - e. (Optional) If the browser supports refresh, go to http://your\_host/server-status?refresh=5 to refresh every five seconds. You will see five requests currently processing 45 idle servers.
  - **Default value:** 250 for IBM HTTP Server V6.0.
  - **Recommended value:** Set this value to prevent bottlenecks, allowing just enough traffic through to the application server.
- **Web server configuration reload interval**
  - **Description:** Tracks a variety of configuration information about WebSphere Application Server resources. The Web server needs to understand some of this information, such as Uniform Resource Identifiers (URIs) pointing to WebSphere Application Server resources. This configuration data is pushed to the Web server through the WebSphere Application Server plug-in at intervals specified by this parameter. Periodic updates add new servlet definitions without having to restart any of the WebSphere Application Server servers. However, the dynamic regeneration of this configuration information is costly in terms of performance. Adjust this parameter in a stable production environment.
  - **How to view or set:** Use the Refresh configuration interval Web server plug-in property to change the current setting for this parameter. In the administrative console, click **Servers > Web Servers > Web\_server\_name > Plug-in properties**.
  - **Default value:** The default reload interval is 60 seconds.
  - **Recommended value:** Increase the reload interval to a value that represents an acceptable wait time between the servlet update and the Web server update.

For more information about the plugin-cfg.xml file see the topic “Web server plug-ins” on page 122.

- **Sun Java System Web server, Enterprise Edition (formerly Sun ONE) - Solaris operating environment**

The default configuration of the Sun ONE Web server, Enterprise Edition provides a single-process, multi-threaded server.

- **Active threads**

- **Description:** Specifies the current number of threads active in the server. After the server reaches the limit set with this parameter, the server stops servicing new connections until it finishes old connections. If this setting is too low, the server can become throttled, resulting in degraded response times. To tell if the Web server is being throttled, consult its perfdump statistics. Look at the following data:
  - **WaitingThreads count:** If WaitingThreads count is getting close to zero, or is zero, the server is not accepting new connections.
  - **BusyThreads count:** If the WaitingThreads count is close to zero, or is zero, BusyThreads is probably very close to its limit.
  - **ActiveThreads count:** If ActiveThreads count is close to its limit, the server is probably limiting itself.



- **How to view or set:** Use the Maximum number of simultaneous requests parameter in the Enterprise Server Manager interface to control the number of active threads within Sun ONE Web server, Enterprise Edition. This setting corresponds to the RqThrottle parameter in the magnus.conf file.
- **Default value:** 512
- **Recommended value:** Increase the thread count until the active threads parameters show optimum behavior.
- **Microsoft Internet Information Server (IIS) - Windows NT and Windows 2000**
  - **IIS permission properties**
    - **Description:** The Web server has several properties that dramatically affect the performance of the application server. The default settings are usually acceptable. However, because other products can change the default settings without user knowledge, make sure to check the IIS settings for the Home Directory permissions of the Web server. The permissions should be set to Script and not to Execute. If the permissions are set to Execute, no error messages are returned, but the performance of WebSphere Application Server is decreased.
    - **How to view or set:** To check or change these permissions, perform the following procedure in the Microsoft management console:
      1. Select the Web site (usually default Web site).
      2. Right-click and select the **Properties** option.
      3. Click the **Home Directory** tab. To set the permissions of the Home Directory:
        - a. In the **Application** settings, select the **Script** check box in the **Permissions** list and clear the **Execute** check box.
        - b. (Optional) Check the permissions of the sePlugin:
          - 1) Expand the Web server.
          - 2) Right-click the sePlugin and select **Properties**.
          - 3) Confirm that the **Execute** permissions are set to **Execute**.
    - **Default value:** Script
    - **Recommended value:** Script
  - **Number of expected hits per day**
    - **Description:** Controls the memory that IIS allocates for connections.
    - **How to view or set:** Using the performance window, set the parameter to More than 100000 in the Web site properties panel of the Microsoft management console.
    - **Default value:** Fewer than 100000
    - **Recommended value:** More than 100000
  - **ListenBackLog parameter**
    - **Description:** Alleviates failed connections under heavy load conditions, if you are using IIS on Windows NT and Windows 2000. Failure typically occurs when you are using more than 100 clients. ListenBackLog increases the number of requests that IIS keeps in its queue. Consider raising this value if you see intermittent Unable to locate server errors in the Netscape browser.
    - **How to view or set:**
      1. Use a command prompt to issue the **regedit** command to access the operating system registry.
      2. In the registry window, locate the parameter in the HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\InetInfo\Parameters\ListenBackLog directory.
      3. Right-click the parameter to adjust the setting according to the server load.
    - **Default value:** 25 (decimal)
    - **Recommended value:** You can set the ListenBackLog parameter can be set as high as 200, without negative impact on performance and with an improvement in load handling.

## Modifying the WebSphere plug-in to improve performance

You can improve the performance of IBM HTTP Server V6.0 (with the WebSphere Web server plug-in) by modifying the plug-in's RetryInterval configuration. The RetryInterval is the length of time to wait before



trying to connect to a server that has been marked temporarily unavailable. Making this change can help the IBM HTTP Server V6.0 to scale higher than 400 users.

The plug-in marks a server temporarily unavailable if the connection to the server fails. Although a default value is 60 seconds, it is recommended that you lower this value in order to increase throughput under heavy load conditions. Lowering the RetryInterval is important for IBM HTTP Server V6.0 on UNIX operating systems that have a single thread per process, or for IBM HTTP Server 2.0 if it is configured to have fewer than 10 threads per process.

How can lowering the RetryInterval affect throughput? If the plug-in attempts to connect to a particular application server while the application server threads are busy handling other connections, which happens under heavy load conditions, the connection times out and the plug-in marks the server temporarily unavailable. If the same plug-in process has other connections open to the same server and a response is received on one of these connections, the server is marked again. However, when you use the IBM HTTP Server V6.0 on a UNIX operating system, there is no other connection since there is only one thread and one concurrent request per plug-in process. Therefore, the plug-in waits for the RetryInterval before attempting to connect to the server again.

Since the application server is not really down, but is busy, requests are typically completed in a small amount of time. The application server threads become available to accept more connections. A large RetryInterval causes application servers that are marked temporarily unavailable, resulting in more consistent application server CPU utilization and a higher sustained throughput.

**Note:** Although lowering the RetryInterval can improve performance, if all the application servers are running, a low value can have an adverse affect when one of the application servers is down. In this case, each IBM HTTP Server V6.0 process attempts to connect and fail more frequently, resulting in increased latency and decreased overall throughput.

---

## Gskit install images files

The Global Security Kit (GSKit) installation image files for the WebSphere Web server plug-ins are packaged on the CD with the Web server plug-in files. You can download the appropriate GSKIT file to the workstation on which your Web server is running. Use the following table to assist you in selecting the correct GSKIT installation image file.

Operating system	GSKit 7 Installation image file
Windows	No image name
AIX	gskta.rte
HP-UX	gsk7bas
Solaris Operating Environment	gsk7bas
Linux	gsk7bas_7.0.3.1.i386.rpm
Linux390	gsk7bas-7.0.3.1.s390.rpm
LinuxPPC	gsk7bas-7.0.3.1.ppc.rpm

---

## Plug-ins: Resources for learning

See this topic in the V6 Information Center to find links links to relevant supplemental information about Web server plug-ins. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

---

## Web server plug-in tuning tips

### Balancing workloads:

During normal operation, the backlog of connections pending to an application server is bound to grow. Therefore, balancing workloads among application servers in a network fronted by a Web server plug-in helps improve request response time.

In a distributed environment, you can limit the number of connections that can be handled by an applications server. To do this:

1. Go to the **Servers > Application Servers >server\_name**.
2. Under Additional Properties, click > **Web Server Plug-in properties** .
3. Select **Set limit** for the Minimum number of connections that can be handled by the Application Server field.
4. Specify in the Connections field the maximum number of connections you want to allow.
5. Then click **Apply** and **Save**.

When this maximum number of connections is reached, the plug-in, when establishing connections, automatically skips that application server, and tries the next available application server. If no application servers are available, an HTTP 503 response code will be returned to the client. This code indicates that the server is currently unable to handle the request because it is experiencing a temporary overloading or because maintenance is being performed.

The capacity of the application servers in the network determines the value you specify for the maximum number of connections. The ideal scenario is for all of the application servers in the network to be optimally utilized. For example, if you have the following environment:

- There are 10 application servers in a cluster.
- All of these application servers host the same applications (that is, Application\_1 and Application\_2).
- This cluster of application servers is fronted by five IBM HTTP Servers.
- The IBM HTTP Servers get requests through a load balancer.
- Application\_1 takes approximately 60 seconds to respond to a request
- Application\_2 takes approximately 1 second to respond to a request.

Depending on the request arrival pattern, all requests to Application\_1 might be forwarded to two of the application servers, say Appsvr\_1 and Appsvr\_2. If the arrival rate is faster than the processing rate, the number of pending requests to Appsvr\_1 and Appsvr\_2 can grow.

Eventually, Appsvr\_1 and Appsvr\_2 are busy and are not able to respond to future requests. It usually takes a long time to recover from this overloaded situation.

If you want to maintain 2500 connections, and optimally utilize the Application Servers in this example, set the number of maximum connections allowed to 50. (This value is arrived at by dividing the number of connections by the result of multiplying the number of Application Servers by the number of Web servers; in this example,  $2500/(10 \times 5) = 50$ .)

Limiting the number of connections that can be established with an application server works best for Web servers that follow the threading model instead of the process model, and only one process is started.

The IBM HTTP Server V6.0.x follows the threading model. To prevent the IBM HTTP Server from starting more than one process, change the following properties in the Web server configuration file (`httpd.conf`) to the indicated values:

```
ServerLimit      1
ThreadLimit     4000
StartServers    1
MaxClients      1024
MinSpareThreads 1
MaxSpareThreads 1024
ThreadsPerChild 1024
MaxRequestsPerChild 0
```

### **Windows** Improving performance in a high stress environment:

If you use the default settings for a Microsoft Windows operating system, you might encounter Web server plug-in performance problems if you are running in a high stress environment. To avoid these problems, consider tuning the the TCP/IP setting for this operating system. Two of the keys setting to tune are `TcpTimedWaitDelay` and `MaxUserPort`.

To tune the `TcpTimedWaitDelay` setting, change the value of the `tcp_time_wait_interval` parameter from the default value of 240 seconds, to 30 seconds:

1. Locate in the Windows Registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\TcpTimedWaitDelay
```

If this entry does not exist in your Windows Registry, create it by editing this entry as a new `DWORD` item.

2. Specify, in seconds, a value between 30 and 300 inclusive for this entry. (It is recommended that you specify a value of 30. )

To tune the `MaxUserPort` setting:

1. Locate in the Windows Registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\MaxUserPort
```

If this entry does not exist in your Windows Registry, create it by editing this entry as a new `DWORD` item.

2. Set the maximum number of ports to a value between 5000 and 65534 ports, inclusive. (It is recommended that you specify a value of 65534,)

See the Microsoft Web site at <http://www.microsoft.com> for more information about these settings.

---

## Tuning Web servers

“Web server tuning parameters” on page 123 lists tuning parameters specific to Web servers. The listed parameters may not apply to all of the supported Web servers. Check your Web server documentation before using any of these parameters.



---

## Chapter 8. Setting up the administrative architecture

After you install and set up the Network Deployment product, 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.

1. Use the settings page for an administrative service to configure administrative services.
2. Configure cells.
3. Configure deployment managers.
4. Manage nodes.
5. Manage node agents.
6. Manage node groups.
7. Configure remote file services.

---

### Cells

*Cells* are logical groupings of one or more nodes in a WebSphere Application Server distributed network.

A cell is a configuration concept, a way for administrators to logically associate nodes with one another. Administrators define the nodes that make up a cell, according to the specific criteria that make sense in their organizational environments.

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.

---

### Configuring cells

When you created a deployment manager profile, a cell was created. A cell provides a way to group one or more nodes of your Network Deployment product. You probably will not need to re-configure the cell. To view information about and 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 drop-down list for **Cell Discovery Protocol** on the settings page for the cell. It is unlikely that you will need to change the cell's protocol configuration from TCP.
3. Click **Custom Properties** and define any name-value pairs needed by your deployment manager.
4. When you installed the WebSphere Application Server Network Deployment product, a node was 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.

**Note:** 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 details, see "IP version considerations for cells" on page 132.

## IP version considerations for cells

Internet Protocol Version 4 is no longer viable for many businesses. Because it is based on 32-bit architecture, there is a growing shortage of Internet Protocol Version 4 (IPv4) addresses. Internet Protocol Version 6 (IPv6) is based on 128-bit architecture, which allows a far greater number of addresses to be available for use over the Internet.

In response, WebSphere Application Server Version 6 now includes support for IPv6, in addition to continued support for IPv4. This means that nodes running WebSphere Application Server Version 6 can use IPv6. (However, note that nodes running WebSphere Application Server Version 5.x cannot use IPv6.)

WebSphere Application Server Version 6 supports a *dual mode* environment in which you can have older legacy applications running on IPv4 and IPv6-enabled applications running on IPv6. Note, however, that there are restrictions on using IPv4 and IPv6 in the same cell. This article documents those restrictions as well as outlines the ways in which you can set up your cells, depending on the version of IP that you will be using.

From an IP perspective, you must adhere to one of the following scenarios:

### Dual mode cell

In a dual mode cell, mixed IPv4 and IPv6 communications are supported. By default, a cell is set to dual mode when it is created. Note, however, that only nodes running WebSphere Application Server Version 6 are valid in a dual mode cell.

IPv4 and IPv6 nodes cannot communicate with each other, so the purpose of the dual mode cell is to enable this communication, thereby allowing you to use your existing applications, running over IPv4, with newer applications that have been enabled for IPv6.

The following illustration shows a dual mode cell:

### IPv4-only cell

In an IPv4-only cell, all nodes must:

- Use IPv4
- Run WebSphere Application Server Version 5.x
- Have host names defined as strings or 32-bit numerical addresses.

It is important to note that, by default, a cell is set to dual mode. However, in order to run in an IPv4-only environment, you will need to explicitly set the cell to IPv4. See the section on JVM settings, in this article, for more information.

**Note:** If you want to run a combination of WebSphere Application Server Version 5.x and WebSphere Application Server Version 6.0 nodes over IPv4, see the section on setting up a *mixed node cell*, below.

### Mixed node cell

A mixed node cell consists of some nodes running WebSphere Application Server Version 5.x and other nodes running WebSphere Application Server Version 6. In a mixed node cell, all nodes must use IPv4. When defining a node that will be used in a mixed node cell, you must specify the host name as a string or as a 32-bit numerical address, regardless of whether the node is running WebSphere Application Server

Version 5.x or WebSphere Application Server Version 6. 128-bit numerical addresses may not be specified.

In a mixed node cell, even though the WebSphere Application Server Version 6 nodes will be configured to use IPv4, the operating system running on them can still support both IPv4 and IPv6. This is true as long as the WebSphere Application Server Version 6 nodes are configured with string-based host names or 32-bit numerical addresses.

Note also, that you can only add Version 5.x nodes into a mixed node cell through migration. You first need to migrate from a Version 5.x Deployment Manager to a Version 6.0 Deployment Manager, and then either keep the Version 5.x nodes or migrate them to Version 6.0 nodes.

### IPv6-only cell

In an IPv6-only cell, all nodes must:

- Use IPv6
- Run WebSphere Application Server Version 6
- Have host names defined as strings or 128-bit numerical addresses.

### Specifying host names

During the installation of WebSphere Application Server, you are asked to provide the host name or IP address of the machine on which the installation is being carried out in the *Host Name or IP address* field. The host name or IP address that you specify is used to advertise this installation to all other WebSphere Application Server installations in the cell configurations. All nodes in the cell will use the host names or IP addresses that are defined in this way to reach each other. In general, it is best to always use a host name to identify a WebSphere Application Server installation. By using a host name, you will not have to be concerned about which IP address is being used (32-bit versus 128-bit), whether it runs on IPv4 or IPv6, and so on. As long as the DNS service is properly configured, the nodes should all be able to work together.

However, if you prefer, you can control which IP stack or address is used. To do this, enter the specific IP address (32-bit for IPv4 or 128-bit for IPv6) into the *Host Name or IP Address* field. This installation will then be identified with this IP address and other WebSphere Application Server nodes will use this IP address to communicate with this node.

When specifying IPv6 addresses, it is good practice to always surround them with protective square brackets. For example, `[fe80::202:57ff:fec4:2334]`. The reason for this is that in system internal processing, IP addresses are often combined with port numbers in the form of `<IP address>:<port number>`, and the colons in IPv6 addresses could be confusing in such circumstances. Note that you cannot use IPv6 addresses, including IPv6 addresses surrounded by square brackets, within the administrative console or the install wizard.

Note that in scripting, the square brackets might have special meaning, depending on the language binding used (for example, Jacl). You can work around this problem by using a special escape character in front of the opening and closing brackets. Using the Jacl binding, for example, the same IPv6 address cited earlier can be entered as `\[fe80::202:57ff:fec4:2334\]`

**Note:** While you cannot use square brackets with IPv6 addresses within the administrative console, you must use square brackets to specify an IPv6 address as part of the administrative console's URL in a browser. This allows the browser to distinguish the IPv6 address from the port value.



## Multicast configuration

WebSphere Application Server uses multicast broadcasting at the node level to allow a node agent to discover the managed processes in the node. IPv4 and IPv6 addresses are not compatible. Therefore, to allow a WebSphere Application Server node installation to run *out-of-the-box*, both IPv4 and IPv6 multicast addresses are initially defined in the node agent configuration, and when a node agent starts, both addresses will be tried in sequence. It is a good idea to delete either `NODE_MULTICAST_DISCOVERY_ADDRESS` or `NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS` after installation. By that time, you should know whether the node is running IPv4 or IPv6, so by limiting multicast discovery to the known protocol, the node agent runs more efficiently.

To delete one of the multicast ports using the Administrative Console, do the following:

1. Click **System Administration --> Node Agents**.
2. Select the node agent.
3. On the next panel, under Additional Properties, select **Ports**. The next panel shows a list of existing ports.
4. Select either `NODE_MULTICAST_DISCOVERY_ADDRESS` (to delete IPv4) or `NODE_IPV6_MULTICAST_DISCOVERY_ADDRESS` (to delete IPv6).
5. Click **Delete**.

## JVM settings

Two system properties are related to IPv6 support. They are:

- `java.net.preferIPv4Stack=<true|false>`
- `java.net.preferIPv6Addresses=<true|false>`

In general, setting these properties is not recommended. In particular, setting `java.net.preferIPv4Stack` to true will disable the dual mode support in the JVM which might, in turn, disrupt normal WebSphere Application Server functions. Therefore, it is important to understand the full implications if you are contemplating using this setting.

## 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.

### 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 cell follows to retrieve information from a network.

Select one of these protocol options:

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

---

## Deployment managers

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.

A deployment manager hosts the administrative console. A deployment manager provides a single, central point of administrative control for all elements of the entire WebSphere Application Server distributed cell. Each cell contains one deployment manager.

---

## Configuring deployment managers

When you created a deployment manager profile, a deployment manager was created. A deployment manager provides a single, central point of administrative control for all elements in a WebSphere Application Server distributed cell. You can run the deployment manager with its default settings. However, you can follow this task to change the deployment manager configuration settings such as the ports the process uses, custom services, logging and tracing settings, and so on. To view information about and manage a deployment manager, use the settings page for a deployment manager.

1. Access the settings page for a deployment manager. Click **System Administration > Deployment Manager** from the navigation tree of the administrative console.
2. Configure the deployment manager as desired by clicking on a property such as **Custom Services** and specifying settings on the resulting pages.

## Running the deployment manager with a non-root user ID

This article describes how to run the deployment manager with a non-root user ID on Linux and UNIX platforms.

If global security is enabled, the user registry must not be Local OS. Using the Local OS user registry requires the dmgr process to run as root. If you are attempting to run a deployment manager as root in WebSphere Application Server Version 6 when you previously used a non-root user ID on Linux and UNIX platforms in Version 5.x, see the *Migrating, coexisting, and interoperating* PDF.

By default, the Network Deployment product on Linux platforms uses the root user to run the deployment manager, which is the dmgr process. You can use a non-root user to run the deployment manager. You might want to change to a non-root user ID for security or administrative reasons.

Perform this task to change the permissions for the deployment manager. Restart the deployment manager for the changes to take effect.

For the steps that follow, assume that:

- wasadmin is the user to run all servers
- wasgroup is the user group
- dmgr is the deployment manager
- the installation root for Network Deployment is *install\_nd\_root*, for example `/opt/IBM/WebSphere/AppServer`
- you created a run-time environment with a single profile or multiple profiles

To configure a user to run the deployment manager, complete the following steps:

1. Log on to the Network Deployment system as root.
2. Create user wasadmin with primary group wasgroup.

3. Start the deployment manager process as root with the `startManager.sh` script.

Issue the script command:

```
network deployment installation root/profiles/deployment manager profile name/bin/
./startManager.sh
```

4. Start the administrative console.
5. Define the `dmgr` process to run as a `wasadmin` process.

Click **System Administration > Deployment manager > Server Infrastructure > Java and Process Management > Process Definition > Additional Properties > Process Execution** and change all of these values:

Property	Value
Run As User	wasadmin
Run As Group	wasgroup
UMASK	022
	where the value 022 means the files the process creates are writable by the group and by others as defined on the Linux or UNIX platforms

6. Save the configuration.
7. Stop the deployment manager with the `stopManager.sh` script.  
Issue the script command from the `network deployment installation root/profiles/profile name/bin` directory:  
`./stopManager.sh`
8. As root, use operating system tools to change file permissions on Linux and UNIX-based platforms. The following example assumes `/opt/IBM/WebSphere/AppServer` is the installation root:

```
chgrp wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name
chgrp wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/config
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/logs
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/wstemp
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/installedApps
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/temp
chgrp -R wasgroup /opt/IBM/WebSphere/AppServer/profiles/profile name/tranlog
chmod g+wr /opt/IBM/WebSphere
chmod g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/config
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/logs
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/wstemp
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/installedApps
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/temp
chmod -R g+wr /opt/IBM/WebSphere/AppServer/profiles/profile name/tranlog
```

9. Log in as `wasadmin` on the Network Deployment system.
10. Start the deployment manager process with the `startManager.sh` script.

Issue the script command:

```
network deployment installation root/profiles/deployment manager profile name/bin/
./startManager.sh
```

You can start a deployment manager process from a non-root user.

## Deployment manager settings

Use this page to stop the deployment manager from running, and to link to other pages which you can use to define additional properties for the deployment manager. A deployment manager provides a single, central point of administrative control for all elements of the entire 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, SAF (for example, RACF), started task control, and others.

The name is 1-8 characters, alpha numeric or national language. It 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

### Process ID

Specifies a string identifying 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 it is not running.

<b>Data type</b>	String
<b>Default</b>	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. Node names usually are identical to the host name for the computer.

Nodes in the network deployment topology can be managed or unmanaged. Two types of managed nodes exist while one type of unmanaged node exists.

One type of managed node has a node agent which manages all servers on a node, whether the servers are WebSphere Application Servers, Java Message Service (JMS) servers (on Version 5 nodes only), Web servers, or generic servers. The node agent represents the node in the management cell. A deployment manager manages this type of managed node. The other type of managed node has no node agent. This type of managed node is defined on a standalone Application Server. The deployment manager cannot manage this standalone Application Server. An standalone Application Server can be federated. When it is federated, a node agent is automatically created. The node becomes a managed node in the cell. The deployment manager manages this node.

An unmanaged node does not have a node agent to manage its servers. Unmanaged nodes can have server definitions such as Web servers in the WebSphere Application Server topology. Unmanaged nodes can never be federated. That is, a node agent can never be added to an unmanaged node.

A supported Web server can be on a managed node or an unmanaged node. You can define only one Web server to a standalone WebSphere Application Server node. This Web server is defined on an unmanaged node. You can define Web servers to the deployment manager. These Web servers can be defined on managed or unmanaged nodes.

WebSphere Application Server supports basic administrative functions for all supported Web servers. For example, 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 created 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.

You can add managed and unmanaged nodes to a network deployment cell in one of the following ways:

- Administrative console
- Command line (managed nodes only)
- Administrative script

- Java program

Each of these methods for adding a managed 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.

Whether you specify an explicit node group or accept the default, the node group membership rules must be satisfied. If the node 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.

---

## Managing nodes

This article 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.

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 manage nodes, use the Nodes page. To access the Nodes page, click **System Administration > Nodes** in the console navigation tree.

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, do one of the following sets of actions 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:
The distributed platform	The distributed platform	Optionally specify a node group and a core group. Click <b>OK</b> .
The distributed 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 <b>OK</b> .
A z/OS system	The distributed 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 <b>OK</b> .

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 Windows 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.

Both Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are now supported by WebSphere Application Server, but restrictions do apply when using both IPv4 and IPv6 in the same cell. When you add a node to a cell, the format in which you specify the name is based on the version of IP that the node is using. For details, see IP version considerations for cells.

- **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**. By default, a node uses Transmission Control Protocol (TCP). You probably not need to change a node protocol configuration from TCP. However, if you do need to change the discovery protocol value, some guidelines are provided:

- For a managed process, use `multicast`. A managed process supports multicast because by using multicasting, all application servers in one node can listen to one port instead of to one port for each server. A benefit of using multicast is that you do not have to configure the discovery port for each application server or prevent port conflicts. A drawback of using multicast is that you must ensure that your machine is connected to the network when application servers (not including the node agent) launch because a multicast address is shared by the network and not by the local machine. If your machine is not connected to the network when application servers launch, the multicast address is not shared with the application servers.
- For a node agent or a deployment manager, use TCP or UDP. Do not use `multicast`.

- **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 added a managed node or changed a managed node's configuration, synchronize the node's configuration. On the Node Agents page, ensure that the node agent for the node is running. Then, on the Nodes page, put a check mark in the check box beside the node whose configuration files you want to synchronize and click **Synchronize** or **Full Resynchronize**.

Clicking either button 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 is important if automatic configuration synchronization is disabled, or if the synchronization interval is set to a long time, and a configuration change has been made to the cell repository that needs to be replicated 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. So 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 will be 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, put a check mark in the check box beside the managed node whose servers you want to stop running and click **Stop**.

- **Remove a node.**

On the Nodes page, put a check mark in the check box beside the node 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 .

## Node collection

Use this page to manage nodes in the WebSphere Application Server environment. Nodes group managed servers.

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.

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 name is based on the version of IP the node will be using.

### Version

Specifies the product version 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 unknown.

### 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)

**multicast**

IP multicast protocol

### Status

Indicates the state of the node.

The state can be **Started**, **Stopped**, **Unavailable**, **Unknown**, **Partial Start**, **Partial Stop**, **Not applicable**, **Synchronized**, or **Not synchronized**.

## 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

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 name is based on the version of IP the node will be using.

### **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 read only. It is defined during installation and customization.

### **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)
- multicast**  
IP multicast protocol

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

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.

## Add managed Windows node

Use this page to run the node agent as a Windows service.

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

***Run the node agent as a Windows service:***

Specifies whether to run a node agent as a Windows service.

**Default** false (cleared)

***User name:***

Specifies the ID for running the service process for the node agent. The user name and password fields are only available if security is enabled. If you do not specify the user name, the node agent runs under the authority of the local system. User name requirements are the requirements that the Windows operating system imposes for a user ID.

***Password:***

Specifies the password for the user name that you supply. Password requirements are the requirements that the Windows operating system imposes for a password.

***Confirm password:***

Specifies the same password that you typed for Password so that you can verify the correct password.

## **Add managed nodes**

A managed node is a node with an Application Server and a node agent that belongs to a cell. Use this page to add a managed node to a cell.

To view this 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.

**Date 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.

<b>Date type</b>	Integer
<b>Default</b>	8880

## 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 notes

- 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.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.

### **com.ibm.websphere.deployed.features**

A list of features that extends a profile. An example of a feature is an administrative console plug-in.

---

## **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.

Nodes that you organize 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 Application Server administrator. However, a node must be a member of a node group. Initially, all Application Server nodes are members of the default node group named DefaultNodeGroup.

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

On the z/OS platform, 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.

A node on a distributed platform and a node on a z/OS platform cannot 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.

## **Node group membership rules**

Nodes can be members of node groups if they meet certain requirements.

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. If the node is on the z/OS platform, the node group must be a sysplex node group.
- Nodes on a distributed platform and nodes on the z/OS platform must be members of different node groups.
- Nodes on the z/OS platform that are in different sysplexes must be members of different groups.

## Examples: Using node groups

Use node groups to define groups of nodes capable of hosting members of the same cluster. An application deployed to a cluster must be capable of running on any of the cluster members. This means that the node that hosts each of the cluster members must be configured with software and settings necessary to support running of 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 1 to 8.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes 6, 7, and 8 are additionally configured as WebSphere Business Integration Server Foundation nodes.
- All nodes are either z/OS system nodes from the same sysplex, or distributed platform nodes.
- By default, all the nodes are in the default node group, DefaultNodeGroup.

Applications that exploit WebSphere Business Integration Server Foundation functions can run successfully only on nodes 6, 7, and 8. Therefore, clusters that host these applications can be formed only on nodes 6, 7, and 8. To define a clustering policy that guides users of your WebSphere cell into building clusters that can only span predetermined nodes, create an additional node group called WBINodeGroup, for example. Add to the node group nodes 6, 7, and 8. 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 1 to 6.
- Each node is a managed node, which means that each node is configured with an Application Server.
- Nodes 1 to 4 are on distributed platforms.
- Nodes 5 and 6 are nodes on the z/OS operating system and are in sysplex PLEX1.
- The deployment manager is on a distributed platform node.
- Nodes 1 to 4 are members of the DefaultNodeGroup node group by default.
- You created empty node group PLEX1NodeGroup 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. You did this because nodes on the z/OS operating system cannot be in the same node group as the distributed platform nodes.

Applications that exploit z/OS functions in the PLEX1 sysplex can run successfully only on nodes 5 and 6. Therefore, clusters that host these applications can be formed only on nodes 5 and 6. The required separation of distributed 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 only span 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

Read about Nodes groups if you are unfamiliar with them.

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.

- 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.

### 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.

On the z/OS platform, 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.

A node on a distributed platform and a node on a z/OS platform cannot 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.

<b>Data type</b>	String
<b>Maximum length</b>	64 characters

The name must contain alphanumeric or national language characters and cannot start with a number.

### **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.

<b>Data type</b>	Integer
------------------	---------

### **Description:**

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

---

## Managing node group members

Read about Nodes groups and Node group membership rules if you are unfamiliar with them.

Group nodes that meet your application requirements into a 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 bread crumb 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 bread crumb 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* > nodes > Node group members**.

### Name

Specifies the name of a node group member.

### Description

Specifies a description that you previously defined for the node group member when you created the 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* > nodes > 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.

<b>Data type</b>	String
<b>Maximum length</b>	64 characters

The name must contain alphanumeric or national language characters and cannot start with a number.

---

## 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**

### Standalone

Specifies whether the server process is a participant in a Network Deployment cell or not. If the box is checked (true), the server does not participate in distributed administration. If the box is unchecked (false), the server participates in the Network Deployment system.

The default value for base WebSphere Application Server installations is true. When addNode runs to incorporate the server into a Network Deployment cell, the value switches to false.

<b>Data type</b>	Boolean
<b>Default</b>	true

### Preferred Connector

Specifies the preferred JMX Connector type. Available options, such as SOAPConnector or RMICConnector, are defined using the JMX Connectors page.

<b>Data type</b>	String
<b>Default</b>	SOAP

## 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***

**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.

**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.

**Name:** The name used to identify the Extension MBean provider library.

## 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* > Extension MBean**

### 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* > Extension MBean > *Extension MBean name***

**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.

## Java Management Extensions connector properties

A Java Management Extensions (JMX) connector can either be a Remote Method Invocation (RMI) connector or a Simple Object Access Protocol (SOAP) connector.

Depending on the property, you can specify or set a property in the administrative console, the wsadmin tool, Application Server commands, scripts 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.

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. For specific information on how to code the JMX connector properties for a custom Java administrative client program, see the "Java API documentation for Application Server" topic in the Information Center.

For the administrative console, this article specifies the coding of the particular setting or property. Coding of properties in the `soap.client.props` file that are specific to JMX connectors is specified. These

properties begin with `com.ibm.SOAP`. Other properties in the `soap.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, is specified.

Each profile has a property file at `installation root/profiles/profile name/properties/soap.client.props`. These property files allow you to set different properties, including security and timeout properties. These properties are the default for all administrative connections that use the SOAP JMX connector between processes executing in a particular profile. For instance, the `wsadmin` program executing under a particular profile uses the property values from that file for the SOAP 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 one of the following paths:

- **Servers -> Application servers -> *server name* -> Server Infrastructure -> Administration -> Administration Services -> Additional properties -> JMX Connectors->*connector type* -> Additional Properties -> Custom properties**
- **System administration -> Deployment manager ->Additional Properties -> Administration Services -> Additional Properties -> JMX Connectors->*connector type*-> Additional Properties -> Custom properties**
- **System administration -> Node agents ->*node agent name* -> Additional Properties -> 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

Specifies the SOAP client 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 level of 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.

You can set the property by using one of the following options:

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

<b>Property</b>	<code>com.ibm.SOAP.requestTimeout</code>
<b>Data type</b>	Integer
<b>Range in seconds</b>	0 to n
<b>Default</b>	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.

<b>Property</b>	<code>requestTimeout</code>
<b>Data type</b>	Integer

<b>Range in seconds</b>	0 to n
<b>Default</b>	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

Specifies the Universal Resource Locator (URL) of the soap.client.props file. Specify the configuration 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.

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

- A Java administrative client. The property is AdminClient.CONNECTOR\_SOAP\_CONFIG.

### Security context provider

Specifies the Secure Sockets Layer (SSL) implementation to use between the Application Server and the SOAP client. You can specify either IBM Java Secure Sockets Extension (IBMJSSE) or IBM Java Secure Sockets Extension that has undergone Federal Information Processing Standards certification (IBMJSSEFIPS).

You can set the property by using the soap.client.props file.

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

### Secure Sockets Layer (SSL) security

Use this property to enable SSL security between Application Server and the SOAP client. You can set the property by using one of the following options:

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

<b>Property</b>	com.ibm.SOAP.securityEnabled
<b>Data type</b>	Boolean
<b>Default</b>	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.

<b>Property</b>	securityEnabled
-----------------	-----------------

<b>Data type</b>	Boolean
<b>Default</b>	False

- A Java administrative client. The property is AdminClient.CONNECTOR\_SECURITY\_ENABLED.

## SOAP and RMI connector properties

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

### Connector type

Specify a connector type of SOAP or RMI, depending on whether Application Server connects to a SOAP server or an RMI 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.

<b>Property</b>	Type
<b>Data type</b>	String
<b>Valid values</b>	SOAPConnector RMIConnector
<b>Default</b>	SOAP

- A Java administrative client. The property is AdminClient.CONNECTOR\_TYPE. Specify by using the AdminClient.CONNECTOR\_TYPE\_RMI or the AdminClient.CONNECTOR\_TYPE\_SOAP constants.

### Host

Use the host property to specify the host name or the IP address of the server to which Application Server connects. The server can be a SOAP server or an RMI 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.

<b>Property</b>	host
<b>Data type</b>	String
<b>Valid values</b>	Host name or IP address
<b>Default</b>	None

- A Java administrative client. The property is AdminClient.CONNECTOR\_HOST.

### Port

Use the port property to specify the port number of the server to which Application Server connects. The server can be a SOAP server or an RMI 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.



<b>Property</b>	port
<b>Data type</b>	Integer
<b>Valid value</b>	Port number
<b>Default</b>	None

- A Java administrative client. The property is AdminClient.CONNECTOR\_PORT.

### User name

Specifies the user name that Application Server uses to access the SOAP server or the RMI 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.

<b>Property</b>	com.ibm.SOAP.loginUserId
<b>Data type</b>	String
<b>Valid value</b>	The value must match the global SSL settings for SOAP or RMI.
<b>Default</b>	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.

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

- A Java administrative client. The property is AdminClient.USERNAME.

### Password

Specifies the password that Application Server uses to access the SOAP server or the RMI 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.

<b>Property</b>	com.ibm.SOAP.loginPassword
<b>Data type</b>	String
<b>Valid values</b>	The value must match the global SSL settings for SOAP or RMI.
<b>Default</b>	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.

<b>Property</b>	password
<b>Data type</b>	String
<b>Valid values</b>	The value must match the global SSL settings for SOAP or RMI.
<b>Default</b>	None

- A Java administrative client. The property is AdminClient.PASSWORD.

## Java Management Extensions connectors

Use this page to view and change the configuration for Java Management Extensions (JMX) connectors.

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.

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.

### Type

Specifies the type of the JMX connector.

<b>Data type</b>	Enum
<b>Default</b>	SOAPConnector
<b>Range</b>	<b>SOAPConnector</b> For JMX connections using Simple Object Access Protocol (SOAP). <b>RMIConnector</b> For JMX connections using Remote Method Invocation (RMI).

## JMX connector settings

Use this page to view the configuration for a Java Management Extensions (JMX) connector.

To view this administrative console page, click one of the following paths:

- **Servers > Application Servers > *server\_name* > Administration > Administration Services > JMX Connectors > *connector\_type***
- **Servers > JMS Servers > *server\_name* > Administration > Administration Services > JMX Connectors > *connector\_type***

### Type:

Specifies the type of the JMX connector.

<b>Data type</b>	Enum
<b>Default</b>	SOAPConnector
<b>Range</b>	<b>SOAPConnector</b> For JMX connections using Simple Object Access Protocol (SOAP). <b>RMIConnector</b> For JMX connections using Remote Method Invocation (RMI).

## 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 Services > Repository Service**.

### Audit Enabled

Specifies whether to audit repository updates in the log file. The default is to audit repository updates.

<b>Data type</b>	Boolean
<b>Default</b>	true

---

## Node agents

Node agents are administrative agents that route administrative requests to servers.

A node agent is a server that runs on every host computer system that participates in the WebSphere Application Server Network Deployment product. It 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.

---

## 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. A node agent is created automatically when a node is added to a cell.

1. View information about a node agent. Use the Node Agents page. 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 on the node agent's name under **Name**.

**Note:** 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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using. For details, see "IP version considerations for cells" on page 132.

2. Stop and then restart the processing of a node agent. On the Node Agents page, place a check mark in the check box beside the node agent you want to restart, then click **Restart**. It is important to keep a node agent running because a node agent must be running in order for application servers on the node managed by the node agent to run.
3. Stop and then restart all application servers on the node managed by the node agent. On the Node Agents page, place a check mark in the check box beside the node agent that manages the node whose servers you want to restart, then click **Restart all Servers on Node**.  
Note that the node agent for the node must be processing (step 2) in order to restart application servers on the node.
4. Stop the processing of a node agent. On the Node Agents page, place a check mark in the check box beside the node agent you want to stop processing, then click **Stop**.

## 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** .

## 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.

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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

## 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.

## Status

Indicates whether the node agent server is started or stopped.

Note that if the status of servers such application servers is *Unavailable*, the node agent is not running in the servers' node and you must restart the node agent before you can start the servers.

## Node agent server settings

Use this page to view information about and 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 in order for the deployment manager node to be able to collect and control servers configured on that node. If you use configuration synchronization support, a node agent coordinates with the deployment manager server to synchronize the node's configuration data with the master copy managed by the deployment manager.

You must initially start a node agent outside the administrative console. For information on how to initially start a node agent, see the WebSphere Application Server Information Center.

The Runtime tab displays only when a node agent runs.

### **Name:**

Specifies a logical name for the node agent server.

### **Data type**

String

### **Default**

NodeAgent Server

**Short name:**

Specifies the short name of the node agent server.

The name is 1-8 characters, alpha-numeric or national language. It 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.

**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.

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 a node is added to a cell, the format in which the name is specified is based on the version of IP the node will be using.

**Data type** String

**State:**

Indicates whether the node agent server is started or stopped.

**Data type** String  
**Default** Started

## 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 file services do the following:

**File transfer service**

The file transfer service enables the moving of files between the network manager and the nodes.

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 defined in the Network Deployment server configuration under its WebContainer HTTP transports.

### 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.

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.

### 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.

<b>Data type</b>	Boolean
<b>Default</b>	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.

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. The default is 1 minute. Increase the time interval to synchronize files less often.

**Data type** Integer  
**Units** Minutes  
**Default** 1

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.

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:

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

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

---

## Administrative agents: Resources for learning

Use the following links to find relevant supplemental information about WebSphere Application Server administrative agents 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 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 at <http://publib-b.boulder.ibm.com/Redbooks.nsf/portals/WebSphere>.

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

- IBM developerWorks WebSphere at <http://www.software.ibm.com/wsdd/>.

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 at <http://www.ibm.com/software/webservers/appserv/support.html>.

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 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 9. Configuring the environment

To assist in handling requests among Web applications, Web containers, and application servers, you can configure cell-widesettings for virtual hosts, variables and shared libraries.

1. Configure virtual hosts.
2. Configure variables.
3. If your deployed applications use shared library files, define the shared library files needed.  
See “Managing shared libraries” on page 175.

---

### Virtual hosts

A virtual host is a configuration that enables a single host machine to resemble multiple host machines. 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.

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 and serve the servlet. If no match is found, an error is returned to the browser.

An application server provides a default virtual host with some common aliases, such as the machine’s IP address, short host name, and fully qualified host name. The alias comprises the first part of the path for accessing a resource such as a servlet. For example, it is localhost:80 in the request `http://localhost:80/myServlet`.

A virtual host is not associated with a particular node (machine). It is a configuration, rather than a “live object,” explaining why you can create it, but cannot start or stop it. For many users, creating virtual hosts is unnecessary because the default\_host is provided.

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.

### Why you would use virtual hosting

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. 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.

Virtual hosts allow the administrator to isolate and independently manage multiple sets of resources on the same physical machine.

Suppose 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.

Further suppose that 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.

The servlets on one virtual host do not share their context with the servlets on the other virtual host. Requests for the servlet on VirtualHost1 can continue as usual. This is true even though VirtualHost2 is refusing to fill requests for the servlet with the same name.

You associate a servlet or other application with a virtual host instead of the actual DNS address.

## The default virtual host (default\_host)

The product provides a default virtual host (named default\_host).

The virtual host configuration uses wildcard entries with the ports for its virtual host entries.

- The default alias is \*:80, using an internal port that is not secure.
- Aliases of the form \*:9080 use the secure internal port.
- Aliases of the form \*:9443 use the external port that is not secure.
- Aliases of the form \*:443 use the secure external port.

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.

## How requests map to virtual host aliases

Virtual hosts let you manage a single application server on a single machine as if the application server were multiple application servers that are each on their own host machine. Resources associated with one virtual host cannot share data with resources associated with another virtual host, even though the virtual hosts share the same application server on the same physical machine.

When you request a resource, WebSphere Application Server tries to map the request to an alias of a defined virtual host.

Mappings are both case sensitive and insensitive. For example, the portion "http://host:port/" is case insensitive, but the URL that follows is case sensitive. The match must be alphanumerically exact. Also, 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 due to 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 was 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 stand-alone WebSphere Application Server, *server1*.
3. The server1 Application Server 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 when WebSphere Application Server 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](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 enable you to isolate and independently manage multiple sets of resources on the same physical machine.

1. Create a virtual host using the “Virtual host collection” of the administrative console. Click **Environment > Virtual Hosts** from the navigation tree of the console. Click **New**. On the “Virtual host settings” on page 169 page that displays, specify an administrative name for the virtual host. When you create a virtual host, a default set of 90 MIME entries is created for the virtual host.
2. There must be a virtual host alias corresponding to each port used by an HTTP transport. There is one HTTP transport in each Web container, usually assigned to the virtual host named default\_host. You can change the default assignment to any valid virtual host.

You must create a virtual host for each HTTP port in the following cases:

- You are using the internal HTTP transport with a port other than the default of 9080. You must define the port that you are using.
- You are using the default port 9080, but the port is no longer defined. You must define port 9080.
- You have created multiple Application Servers (either stand-alone servers or cluster members) that use the same virtual host. Because each server must be listening on a different HTTP transport port, you must define a virtual host alias for the transport port of each server.

If you define new virtual host aliases, identify the port values that the aliases use on the “HTTP transport collection” on page 217 page.

3. If necessary, create a virtual host alias for each HTTP transport port.  
From the “Virtual host collection” page, click your virtual host. On the “Virtual host settings” on page 169 page for the virtual host, click **Host aliases**. To define a virtual host alias on the “Host alias collection” on page 169 page, click **New**. On the “Host alias settings” on page 170 page for the virtual host alias, specify a host name and a port. Configure the virtual host to contain an alias for the port number. For example, specify an alias of \*:9082 if 9082 is the port number in use by the transport.
4. When you enter the URL for the application into a Web browser, include the port number in the URL.  
For example, if 9082 is the port number, specify a URL such as  
`http://localhost:9082/wlm/SimpleServlet`
5. If MIME entries are not specified at the Web module level, define MIME object types and their file name extensions. For each needed MIME entry on the “MIME type collection” on page 171 page, click **New**. On the “MIME type settings” on page 171 page, specify a MIME type and extension.
6. After you configure a virtual host alias or change a configuration, you must regenerate the Web server plug-in configuration and restart WebSphere 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 (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 > Virtual Hosts > 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.

<b>Data type</b>	String
<b>Default</b>	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 Hosts > 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.

The default reflects the value specified at product setup. The default might be 80, 81, 9060 or a similar value.

**Data type**

Integer

**Default**

9060

## 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

---

## Variables

A variable is a configuration property that can be used to provide a parameter for some values in the system. A variable has a name and a value.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA\_HOME, and APP\_INSTALL\_ROOT.

- Configuring certain cell-wide customization values.

Each variable has a scope. A scope is the range of locations in the WebSphere Application Server network where the variable is applicable.

- A variable with a cell-wide scope is available across the entire WebSphere Application Server 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.

You can use variables in configuration values such as file system path settings. Use the following syntax to refer to a variable:

```
${variable_name}
```

The value of a variable can contain a reference to another variable. The value of the variable is computed by substituting the value of the referenced variable recursively.

Variables are useful when concatenating two path variables when the specification does not accept the *AND* operator. For example, suppose that the following variables exist:

Variable name	Variable value
<b>ROOT_DIR</b>	/
<b>HOME_DIR</b>	\${ROOT_DIR}home
<b>USER_DIR</b>	\${HOME_DIR}/myuserdir

The variable reference `${USER_DIR}` resolves to the value `/home/myuserdir`.

---

## Configuring WebSphere variables

This topic describes how to create a WebSphere Application Server variable.

You can define a WebSphere Application Server variable to provide a parameter for some values in the system. After you define the name and value for a variable, the value is used in place of the variable name. Variables most often specify file paths. However, some system components also support the use of variables.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as `JAVA_HOME`, and `APP_INSTALL_ROOT`.
- Configuring certain cell-wide customization values.

The scope of a variable can be cell-wide, node-wide, or applicable to only one server process.

Define variables on the **Environment > WebSphere Variables** console page.

Define the scope to apply a variable cell-wide, cluster-wide, node-wide or to only one server process. A variable resolves to its new value when used in a component that supports the use of variables.

1. Click **Environment > WebSphere Variables** in the administrative console to define a new variable.
2. Specify the scope of the variable. Declare the new variable for the **Cell**, **Cluster**, **Node** or **Server** and click **Apply**.

The variable exists at the level you specify. Define a variable at multiple levels to use multiple values. The more granular definition overrides the higher level setting.

For instance, if you specify the same variable on a node and a server, the server setting overrides the node setting. Similarly, a node level setting overrides a cluster or cell setting.

Scoping variables is particularly important when 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.

See the *Developing and deploying applications* PDF for more information.

3. Click **New** on the WebSphere Variables page.
4. Specify a name, a value, and a description on the Variable page. Click **OK**.
5. Verify that the variable is displayed in the list of variables. The administrative console does not pick up typing errors. The variable is ignored if it is referred to incorrectly.
6. Save your configuration.
7. Stop the server and start the server again to put the variable configuration into effect.

## WebSphere variables collection

Use this page to view and change a list of substitution variables with their values and scope.

To view this administrative console page, click **Environment > WebSphere Variables**.

For information on a variable, click the variable and read the value in the **Description** field.

### 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 Application Server variable is visible on the administrative console panel.

A resource can be visible in the administrative console collection table at the cell, cluster, node, or server scope.

## Variable settings

Use this page to define the name and value of a WebSphere Application Server substitution variable.

To view this administrative console page, click **Environment > WebSphere Variables > WebSphere\_variable\_name**.

### Name:

Specifies the symbolic name for a WebSphere Application Server variable. For example, a variable name might represent a physical path or URL root that is used by WebSphere Application Server.

WebSphere variables are used for:

- Configuring WebSphere Application Server path names, such as JAVA\_HOME, and APP\_INSTALL\_ROOT.
- Configuring certain cell-wide customization values.

WebSphere Application Server substitutes the symbolic name wherever its value displays in the system.

For example, "JAVA\_HOME" is the symbolic name representing the file system path to the installation directory for the Java Virtual Machine (JVM). For example, the value is /opt/IBM/WebSphere/AppServer/java for the WebSphere Application Server product on a Linux machine.

You can create new variables for use in WebSphere Application Server components that support the use of variables.

**Data type** String

**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.

For example, /opt/IBM/WebSphere/AppServer/java is the value on a Linux machine for a variable named JAVA\_HOME.

**Data type** String

**Description:**

Documents the purpose of a variable.

**Data type** String

## IBM Toolbox for Java JDBC driver

WebSphere Application Server supports the **IBM Toolbox for Java** JDBC driver. The IBM Toolbox for Java JDBC driver is included with the IBM Toolbox for Java product.

IBM Toolbox for Java is a library of Java classes that are optimized for accessing iSeries and AS/400 data and resources. You can use the IBM Toolbox for Java JDBC driver to access local or remote **DB2 UDB for iSeries 400** databases from server-side and client Java applications that run on any platform that supports Java.

IBM Toolbox for Java is available in these versions:

### IBM Toolbox for Java licensed program

The licensed program is available with every OS/400 release, starting with Version 4 Release 2 (V4R2). You can install the licensed program on your iSeries 400 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 iSeries information center: <http://publib.boulder.ibm.com/iseries/v5r1/ic2924/index.htm>. 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-1.ibm.com/servers/eserver/iseries/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 JDBC driver for both versions supports JDBC 2.0. For more information about IBM Toolbox for Java and JTOpen, see the product Web site at <http://www-1.ibm.com/servers/eserver/iseries/toolbox/index.html>.

**Note:** If you are using WebSphere Application Server on platforms other than iSeries, use the **JTOpen** version of the Toolbox JDBC driver.

## Configure and use the jt400.jar file

1. Download the *jt400.jar* file from the **JTOpen** URL at <http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm>. Place it in a directory on your workstation such as *C:\JDBC\_Drivers\Toolbox*.
2. Open the administrative console.
3. Select **Environment**.
4. Select **Managed WebSphere Variables**.
5. Set the managed variable *OS400\_TOOLBOX\_JDBC\_DRIVER\_PATH* at the **Node** level.
6. Double click **OS400\_TOOLBOX\_JDBC\_DRIVER\_PATH**.
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,  

```
OS400_TOOLBOX_JDBC_DRIVER_PATH == "C:\JDBC_Drivers\Toolbox"
```

When you choose a Toolbox driver from the list of possible resource providers the **Classpath** field looks like:

```
Classpath == ${OS400_TOOLBOX_JDBC_DRIVER_PATH}/jt400.jar
```

---

## Shared library files

Shared library files in WebSphere Application Server consist of a symbolic name, a Java class path, and a native path for loading Java Native Interface (JNI) libraries.

You can define a shared library at the cell, node, or server level. Defining a library at one of the three levels does not cause the library to be placed into the application server's class loader. You must associate the library to an application or server in order for the classes represented by the shared library to be loaded in either a server-wide or application-specific class loader.

A separate class loader is used for shared libraries that are associated with an application server. This class loader is the parent of the application class loader, and the WebSphere Application Server extensions class loader is its parent. Shared libraries that are associated with an application are loaded by the application class loader.

---

## Managing shared libraries

Shared libraries are files used by multiple applications. Using the administrative console, you can define a shared library at the cell, node, or server level. You can then associate the library to an application or server to load the classes represented by the shared library in either a server-wide or application-specific class loader. Using an installed optional package, you can 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.

If your deployed applications use shared library files, define shared libraries for the library files and associate the libraries with specific applications or with an application server. Associating a shared library file with a server associates the file with all applications on the server. Use the Shared Libraries page to define new shared library files to the system and remove them.

- Use the administrative console to define a shared library.
  1. Create a shared library for each library file that your applications need.
  2. Associate each shared library with an application or a server.
    - Associate a shared library with an application that uses the shared library file.



- 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.

The first step for making a library file available to multiple applications deployed on a server is to create a shared library for each library file that your applications need. When you create the shared libraries, set variables for the library 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. Change the scope of the collection table to see what shared libraries are in a cell, node, or server.
  - a. Select the cell, a node, or a server.
  - b. Click **Apply**.
3. Click **New**.
4. Configure the shared library.
  - a. On the settings page for a shared library, specify the name, class path, and any other variables for the library file that are needed.
  - b. Click **Apply**.
5. Repeat steps 1 through 4 until you define a shared library instance for each library file that your applications need.

Using the administrative console, associate your shared libraries with specific applications 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.

Alternatively, you can use an installed optional package to associate your shared libraries with an application.

## 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**.

By default, a shared library is accessible to applications deployed (or installed) on the same node as the shared library file. Use the **Scope** field to change the scope to a different node or to a specific server.

### 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***.

### **Name:**

Specifies a name for the shared library.

**Data type** String

### **Description:**

Describes the shared library file.

**Data type** String

### **Classpath:**

Specifies the class path used to locate the JAR files for the shared library support.

**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.

**Data type** String  
**Units** Class path

## Associating shared libraries with applications

You can associate a shared library with an application. Classes represented by the shared library are then loaded in the application's class loader, making the classes available to the application.

This article assumes that you have defined a shared library at the cell, node, or server level. The shared library represents a library file used by multiple deployed applications.

This article also assumes that you want to use the administrative console, and not an installed optional package, to associate a shared library with an application.

To associate a shared library with an application, 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. Click **Applications > Enterprise Applications > *application\_name* > Libraries** in the console navigation tree to access the Library Ref page.
2. Click **Add**.
3. On the settings page for a library reference, specify variables for the library reference as needed. The variables identify the shared library file that your application uses.

#### 4. Click **Apply**.

The name of the library reference is shown in the list on the Library Ref page.

Repeat steps 2 through 4 until you define a library reference instance for each shared library that your application requires.

## 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.

This article assumes that you have defined a shared library at the cell, node, or server level. The shared library represents a library file used by multiple deployed applications.

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 > Application Servers > *server\_name*** to access the settings page for the application server.
  - 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” on page 196 and class loaders.
2. Create a library reference for each shared library file that your application needs.
  - a. Go to the settings page for a class loader.
  - b. Click **Libraries** to access the Library Ref page.
  - c. Click **Add**.
  - d. On the settings page for a library reference, specify variables for the library reference as needed. The variables identify 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 Ref page.

Repeat the previous steps until you define a library reference for each shared library that your application needs.

## 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 2 Platform, Enterprise Edition (J2EE) application is installed on a server or cluster, dependency information is specified in its manifest file. WebSphere Application Server reads the dependency information of the application (.ear file) to automatically associate the application with an installed optional package .jar file. WebSphere Application Server 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 WebSphere Application Server are described in section 8.2 of the J2EE specification, Version 1.4 at [http://java.sun.com/j2ee/j2ee-1\\_4-fr-spec.pdf](http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf).

WebSphere Application Server supports using the manifest file (manifest.mf) in shared library .jar files and application .ear files. 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.

### 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 J2EE application or a module within a J2EE 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.

Read about installed optional packages in "Installed optional packages" on page 178 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](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.

Installed optional packages expand the existing shared library capabilities of an application server. Prior to Version 6, 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 WebSphere Application Server 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.
5. Install the application on the server or cluster.

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 **Applications > Enterprise Applications > *application\_name* > Libraries**.

If no shared libraries are defined, 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 can be used by deployed applications. 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 **Applications > Enterprise Applications > *application\_name* > Libraries > *library\_reference\_name***. A shared library must be defined to view this page.

A shared library is a container-wide library file that can be used by deployed applications. 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

---

**Environment: Resources for learning**

Use the following links to find relevant supplemental information about configuring the WebSphere Application Server cell-wide environment. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

**Programming instructions and examples**

- WebSphere Application Server education at <http://www.ibm.com/software/websphere/technical>.

**Administration**

- Listing of all IBM WebSphere Application Server Redbooks at <http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere>.



---

## Chapter 10. Working with server configuration files

Application server configuration documents define the available application servers, their configurations, and their contents.

You should periodically save changes to your administrative configuration. You can change the default locations of configuration files, as needed.

1. Edit configuration files. The master repository is comprised of .xml configuration files. You can edit configuration files using the administrative console, scripting, wsadmin commands, programming, or by editing a configuration file directly.
2. Save changes made to configuration files. Using the console, you can save changes as follows:
  - a. Click **Save** on the taskbar of the administrative console.
  - b. Put a check mark in the **Synchronize changes with Nodes** check box.
  - c. On the Save page, click **Save**.
3. Handle temporary configuration files resulting from a session timing out.
4. Change the location of temporary configuration files.
5. Change the location of backed-up configuration files.
6. Change the location of temporary workspace files.
7. Back up and restore configurations.

---

### Configuration documents

WebSphere Application Server stores configuration data for servers in several documents in a cascading hierarchy of directories. The configuration documents describe the available application servers, their configurations, and their contents. Most configuration documents have XML content.

#### 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*.

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 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
```

## 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" states whether you can edit a document using the administrative tools or must edit it directly.

---

## Configuration document descriptions

Most configuration documents have XML content. The table below 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

(Locations split for publishing)

Configuration file	Locations	Purpose	Manual editing required
admin-authz.xml	config/cells/ <i>cell_name/</i>	Define a role for administrative operation authorization.	X
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.	
cluster.xml	config/cells/ <i>cell_name/</i> clusters/ <i>cluster_name/</i>	Identify a cluster and its members and weights.  This file is only available with the Network Deployment product.	
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
multibroker.xml	config/cells/ <i>cell_name/</i>	Configure a data replication message broker.	
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/ <i>cell_name/</i> nodes/ <i>node_name/</i>	Identify a node.	
pmirm.xml	config/cells/ <i>cell_name/</i>	Configure PMI request metrics.	X

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

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)

---

## Configuration repositories

A configuration repository stores configuration data. By default, configuration repositories reside in the *config* subdirectory of the product installation 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 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.

---

## 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.

When a session times out, the configuration file in use is saved under the *userid/timeout* directory under the ServletContext's temp area. This is the value of the *javax.servlet.context.tempdir* attribute of the ServletContext. By default, it is: *install\_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 next time you log on to the console, you are prompted to load the saved configuration file. If you decide to load the saved file:

1. If a file with the same name exists in the *install\_root/config* directory, that file is moved to the *userid/backup* directory in the temp area.
2. The saved file is moved to the *install\_root/config* directory.
3. The file is then loaded.

If you decide not to load the saved file, it is deleted from the *userid/timeout* directory in the temp area.

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.

---

## Changing the location of temporary configuration files

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 using system variables or the administrative console.

The default location for the configuration temporary directory is `CONFIG_ROOT/temp`. Change the location by doing either of the following:

- Set the system variable `was.repository.temp` to the location you want for the repository temporary directory. Set the system variable when launching a Java process using the `-D` option. For example, to set the default location of the repository temporary directory, use the following option:  
`-Dwas.repository.temp=%CONFIG_ROOT%/temp`
- Use the administrative console to change the location of the temporary repository file location for each server configuration. For example, on the Network Deployment product, to change the setting for a deployment manager, do the following:
  1. Click **System Administration > Deployment Manager** in the navigation tree of the administrative console. Then, click **Administration Services, Repository Service, and 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 can include WebSphere Application Server variables such as `${WAS_TEMP_DIR}/config`. Then, click **OK**.

The system property set using the first option takes precedence over the configuration property set using the second option.

---

## Changing the location of backed-up configuration files

During administrative processes like adding a node to a cell or updating a file, configuration files are backed up to a backup location. The default location for the backup configuration directory is `CONFIG_ROOT/backup`. Change the location by doing either of the following:

- Set the system variable `was.repository.backup` to the location you want as the repository backup directory. Set the system variable when launching a Java process using the `-D` option. For example, to set the default location of the repository backup directory, use the following option:  
`-Dwas.repository.backup=%CONFIG_ROOT%/backup`
- Use the administrative console to change the location of the repository backup directory for each server configuration. For example, on the Network Deployment product, do the following to change the setting for a deployment manager:
  1. Click **System Administration > Deployment Manager** in the navigation tree of the administrative console. Then, click **Administration Services, Repository Service, and 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 can include WebSphere Application Server variables such as `${WAS_TEMP_DIR}/backup`. Then, click **OK**.

The system property set using the first option takes precedence over the configuration property set using the second option.

---

## Changing the location of temporary workspace files

The administrative console workspace allows client applications to navigate the configuration. Each workspace has its own repository location defined either in the system property or the property passed to a workspace manager when creating the workspace: `workspace.user.root` or `workspace.root`, which is calculated as `%workspace.root%/user_ID/workspace/wstemp`.

The default workspace root is calculated based on the user installation root: `%user.install.root%/wstemp`. You can change the default location of temporary workspace files by doing the following:

- Distributed platforms: Change the setting for the system variable `workspace.user.root` or `workspace.root` so its value is no longer set to the default location. Set the system variable when launching a Java process using the `-D` option. For example, to set the default location the full path of the root of all users' directories, use the following option:

```
-Dworkspace.user.root=full_path_for_root_of_all_user_directories
```

---

## Backing up and restoring administrative configurations

WebSphere Application Server represents its administrative configurations as XML files. You should back up configuration files on a regular basis.

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.
3. Run the `restoreConfig` command to restore configuration files. Specify backup files that do not contain invalid or inconsistent configurations.

---

## 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

---

## 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 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 at <http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere>.



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

- IBM developerWorks WebSphere at <http://www.software.ibm.com/wsdd>.

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 at <http://www.ibm.com/software/webservers/appserv/support.html>.

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-3.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 11. Administering application servers

An application server configuration provides settings that control how an application server provides services for running enterprise applications and their components.

This section describes how to create and configure application servers in an existing application server environment.

A WebSphere Application Server administrator can configure one or more application servers and perform tasks such as the following:

1. Create application servers.
2. Manage application servers.
3. Configure transport chains.
4. Develop custom services.
5. Define processes for the application server. As part of defining processes, you can define:
6. Use the Java virtual machine.

After preparing a server, deploy an application or component on the server. See “Preparing to host applications” on page 251 for a sample procedure that you might follow in configuring the application server runtime and resources.

---

### Application servers

Application servers extend a Web server’s capabilities to handle Web application requests, typically using Java technology. An application server makes it possible for a server to generate a dynamic, customized response to a client request.

For example, suppose--

1. A user at a Web browser on the public Internet visits a company Web site. The user requests to use an application that provides access to data in a database.
2. The user request flows to the Web server.
3. The Web server determines that the request involves an application containing resources not handled directly by the Web server (such as servlets). It forwards the request to a WebSphere Application Server product.
4. The WebSphere Application Server product forwards the request to one of its application servers on which the application is running.
5. The invoked application then processes the user request. For example:
  - An application servlet prepares the user request for processing by an enterprise bean that performs the database access.
  - The application produces a dynamic Web page containing the results of the user query.
6. The application server collaborates with the Web server to return the results to the user at the Web browser.

The WebSphere Application Server product provides multiple application servers that can be either separately configured processes or nearly identical clones.

### Understanding server templates

One step in the process of creating an application server is to specify a template. A server template is used to define the configuration settings of the new server. You have the option of specifying the default server template or choosing a template that is based on a server that already exists. The default template will be used if you do not specify a different template when you create the server.

For information about creating server templates, see “Creating server templates” on page 193. For information about listing server templates, see “Listing server templates” on page 193. For information about deleting server templates, see “Deleting server templates” on page 194.

---

## Creating application servers

For the Network Deployment and z/OS products, you can create a new application server using either the **createApplicationServer**, **createWebServer**, or the **createGenericServer** wsadmin commands (see the *Administering applications and their environment* PDF), or the **Create New Application Server** wizard in the administrative console. You can also create a new application server when you add a cluster member to a server cluster.

With WebSphere Application Server Version 6.0, you can now upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you may be managing servers that are at the current release and servers that are running the newer release in the same cell. In this mixed environment, there are restrictions on what you can do with servers at the older release level. They are:

- You can only create new server definitions on nodes that are running WebSphere Application Server Version 6.0.
- When you create a new server definition, you must use a server configuration template, and that template must be created from a WebSphere Application Server Version 6.0 server instance. You cannot create (or use) a template from a WebSphere Application Server Version 5.x server instance.

There are no restrictions on what you can do with the servers running on the newer release level.

The steps below describe how to use the Create New Application Server page.

1. Go to the Application Servers page and click **New**. This brings you to the Create New Application Server page. You can also create the new application server using the wsadmin **createApplicationServer** command. For information, see Commands for the AdminTask object.
2. Follow the instructions on the Create New Application Server page and define your application server.
  - a. Select a node for the application server.
  - b. Type in a name for the application server. The name must be unique within the node.
  - c. Select a template to be used in creating the new server. You can use a default application server template for your new server or use an existing application server as a template. The new application server will inherit all properties of the template server.
  - d. Select whether the new server will have unique ports for each HTTP transport. By default, this option is enabled. If you select this option, 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, ensure that the default port values do not conflict with other servers on the same physical machine.
  - e. If you create the new server using an existing application server as a model, select whether to map applications from the existing server to the new server. By default, this option is disabled.
3. To use multiple language encoding support in the administrative console, configure an application server with UTF-8 encoding enabled.

The new application server appears in the list of servers on the Application Servers page.

Note that the application server created has many default values specified for it. An application server has many properties that can be set and creating an application server on the Create New Application Server page specifies values for only a few of the important properties. To view all of the properties of your application server and to customize your application server further, click on the name of your application server on the Application Servers page and change the settings for your application server as needed.

## Configuring application servers for UTF-8 encoding

To use multiple language encoding support in the administrative console, you must configure an application server with UTF-8 encoding enabled.

1. Create an application server or use an existing application server.
2. On the Application Server page, click on the name of the server you want enabled for UTF-8.
3. On the settings page for the selected application server, under Server Infrastructure, click **Java and Process Management > Process Definition**.
4. On the Process Definition page, click **Java Virtual Machine**.
5. On the Java Virtual Machine page, specify `-Dclient.encoding.override=UTF-8` for **Generic JVM Arguments** and click **OK**.
6. Click **Save** on the console task bar.
7. Restart the application server.

Note that the `autoRequestEncoding` option does not work with UTF-8 encoding enabled. The default behavior for WebSphere Application Server is, first, to check if `charset` is set on content type header. If it is, then the product uses content type header for character encoding; if it is not, then the product uses character encoding set on server using the system property `default.client.encoding`. If `charset` is not present and the system property is not set, then the product uses ISO-8859-1. Enabling `autoRequestEncoding` on a Web module changes the default behavior: if `charset` is not present on an incoming request header, the product checks the `Accept-Language` header of the incoming request and does encoding using the first language found in that header. If there is no `charset` on content type header and no `Accept language` header, then the product uses character encoding set on server using the system property `default.client.encoding`. As with the default behavior, if `charset` is not present and the system property is not set, then the product uses ISO-8859-1.

## Creating server templates

The steps below describe how to create a server template.

1. In the administrative console, go to **Servers --> Application Servers -->** and then click **Templates**. This brings you to the *Server templates* page.

Note that you can also create server templates using the `createServerTemplate` wsadmin command. For more information, see the *Administering applications and their environment* PDF.

2. On the Server templates page, click **New**. This brings you to the *Select a server* page.
3. Select the server that you want to use to create the new template, then click **OK**. This brings you to the *Create new server template* page.
4. On the *Create new server template* page, enter the name of the new template and, optionally, a description, then click **OK**. This brings you to the *Server templates* page. You should see your new template on the list.

## Listing server templates

The step below describes how to list your server templates.

In the administrative console, go to **Servers --> Application Servers -->** and then click **Templates**. This brings you to the *Server templates* page, which lists all of the existing templates.

Note that you can also list server templates using the `listServerTemplates` wsadmin command. For more information, see the *Administering applications and their environment* PDF.

## Deleting server templates

The steps below describe how to delete a server template.

1. In the administrative console, go to **Servers** --> **Application Servers** --> and then click **Templates**. This brings you to the *Server templates* page.

Note that you can also delete server templates using the **deleteServerTemplate** wsadmin command. For more information, see the *Administering applications and their environment* PDF.

2. On the *Server templates* page, select the template you want to delete, then click **Delete**. The template you chose is removed from the list.

---

## Managing application servers

To view information about an application server, use the Application Servers panel on the administrative console.

You can use the **Application Servers** panel of the administrative console, scripting, or command line tools such as `startServer` and `stopServer` to manage your application servers.

**Note:** With WebSphere Application Server Version 6.0, you can now upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you may be managing servers that are at the current release and servers that are running the newer release in the same cell. Note that in this mixed environment, there are some restrictions on what you can do with servers that are running the older release level. There are no restrictions on what you can do with the servers that are running on the newer release level. For details, see the *Administering applications and their environment* PDF and “Creating clusters” on page 264.

1. Access the Application Servers page. Click **Servers > Application Servers** in the console navigation tree.
2. View information about application servers.

The Application Servers page lists application servers in the cell and the nodes holding the application servers. The Network Deployment product also shows the status of the application servers. The status indicates whether a server is started, stopped, or unavailable.

To view additional information about a particular application server or to further configure an application server, click on the application server name under **Name**. This accesses the settings page for an application server.

To view product information for an application server:

- a. Verify that the application server is running.
- b. Display the **Runtime** tab on the settings page for an application server.
- c. Click **Product Information**.

The Product Information page displayed lists the WebSphere Application Server products installed for the application server, the version and build levels for the products, the build dates, and any interim fixes applied to the application server.

**Note:** You can also get this information by using the **versionInfo** command. For more information, see the *Installing your application serving environment* PDF.

3. Create an application server.
  - a. Click **Servers > Application Servers** in the console navigation tree to access the Application Servers page.
  - b. Click **New**, and follow the instructions on the Create New Application Server page.
4. Start your application server.
5. Monitor the running of application servers.

6. Stop your application server.
7. Delete an application server.
  - a. Click **Servers > Application Servers** in the console navigation tree to access the Application Servers page.
  - b. Place a checkmark in the check box beside an application server to delete it.
  - c. Click **Delete**.
  - d. Click **OK** to confirm the deletion.

## Server collection

Use this page to view information about and manage application servers, generic servers, Java Message Service (JMS) servers, and Web servers.

### Application Servers

The Application Servers page lists the application servers in the cell. You can use this page to create new application servers, create application server templates, or delete existing application servers. You can also use this page to start and stop these application servers.

The Network Deployment product also shows the status of the application servers. The status indicates whether a server is running, stopped, or encountering problems.

To view this administrative console page, click **Application Servers**.

### Generic Servers

The Generic Servers page lists the generic servers in the cell. You can use this page to create new generic servers, create generic server templates, or delete existing generic servers. You can also use this page to start and stop these generic servers.

The Network Deployment product also shows the status of the generic servers. The status indicates whether a server is running, stopped, or encountering problems.

You can use this page to add or delete application servers.

To view this administrative console page, click **Generic Servers**.

### Java Message Service (JMS) Servers

The JMS Servers page lists the JMS servers in the cell. You can use this page to start and stop these JMS servers.

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 running WebSphere Application Server 6.0, but the existing Version 5.x JMS servers will continue to be displayed, and you can modify their properties. You can also delete Version 5.x JMS servers.

To view this administrative console page, click **JMS Servers**.

### Web Servers

The Web Servers page lists the Web servers in your administrative domain. You can use this page to generate and propagate a Web server plug-in configuration file, create new Web servers, create new Web server templates, or delete existing Web servers. You can also use this page to start and stop these Web servers.

To view this administrative console page, click **Web Servers**.

### **Name**

Specifies a logical name for the server. For WebSphere Application Server, server names must be unique within a node.

### **Node**

Specifies the name of the node holding the server.

### **Version**

Specifies the version of the WebSphere Application Server product on which the server runs.

### **Status**

Indicates whether the server is started or stopped. (Network Deployment only)

Note that 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.

### **Application server settings**

An application server is a server which provides services required to run enterprise applications. Use this page to view or change the settings of an application server instance.

To view this administrative console page, click **Servers > Application 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 server. Server names must be unique within a node. However, for multiple nodes within a cluster, you may have different 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. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

<b>Data type</b>	String
<b>Default</b>	server1

#### ***Run in development mode:***

Enabling this option may reduce the startup time of an application server. This may include JVM settings such as disabling bytecode verification and reducing JIT compilation costs. Do not enable this setting on production servers. This setting is only available on application servers running WebSphere Application Server Version 6.0 and later.

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 will not be started in development mode. Setting this option to `true` specifies that the server will be started in development mode (with settings that will speed server startup time).

<b>Data type</b>	Boolean
<b>Default</b>	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 the server be started using multiple threads. Setting this option to `false` specifies that the server will not be started in using multiple threads (which may lengthen startup time).

Note that the order in which the applications start depends on the weights you assigned to each them. Applications that have the same weight are started in parallel. You set an application's weight with the *Starting weight* option on the **Applications > Enterprise Applications > application\_name** page of the Administrative Console. For more information about the *Starting weight* option, see the *Installing your application serving environment* PDF.

<b>Data type</b>	Boolean
<b>Default</b>	true

#### ***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 should search in the parent class loader or in the application class loader first to load a class. The standard for Developer Kit class loaders and WebSphere Application Server class loaders is `Parent first`.

If you select `Parent last`, 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 native operating system's process ID for this server.

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

#### ***Cell name:***

The name of the cell in which this server is running.

The Cell name property is read only.

#### ***Node name:***

The name of the node in which this server is running.

The Node name property is read only.

**State:**

The run-time execution state for this server.

The State property is read only.

**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 > Application Servers > *server\_name* Communications > Ports**.

Note that 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.

For WebSphere Application Server for Network Deployment, you can view this administrative console page by clicking one of the following paths:

- **Servers > Application Servers > *server\_name* > Ports > *port\_name***
- **Servers > JMS Servers > *server\_name* > Security Port Endpoint**
- **Servers > JMS Servers > *server\_name* > Ports > *port\_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 a radio button to:

**Well-known Port**

select a previously defined port from the drop down list

**User-defined Port**

create a port with a new name by entering the name in the text box

**Data type** String

*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.

**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.

**Data type** Integer  
**Default** None

Range	1-65536
-------	---------

**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 a **Custom Properties** link.

*Name:*

Specifies the name (or key) for the property.

Do not start your property names with was. because this prefix is reserved for properties that are predefined in WebSphere 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.

For Network Deployment and the z/OS platform, you can view this administrative console page by clicking one of the following paths:

- **Servers > Application Servers > *server\_name***. Then, under Server Infrastructure, click **Administration > Custom Properties**
- **Servers > JMS Servers > *server\_name***. Then, under Server Infrastructure, click **Administration > Custom Properties**

*Name:*

Specifies the name (or key) for the property.

Do not start your property names with `was.` because this prefix is reserved for properties 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

***Server component collection:***

Use this page to view information about and manage server component types such as application servers, messaging servers, or name servers.

To view this administrative console page, click **Servers > Application Servers > *server\_name***. Then, under Server Infrastructure, click **Administration > Server Components**.

*Type:*

Specifies the type of internal server.

*Server component settings:*

Use this page to view or configure a server component instance.

To view this administrative console, click **Servers > Application Servers > *server\_name***. Then, under Server Infrastructure, click **Administration > Server Components > *server\_component\_name***.

*Name:*

Specifies the name of the component.

<b>Data type</b>	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*.

<b>Data type</b>	String
<b>Default</b>	Started

***Thread pool collection:***

Use this page to select or create a group of threads that an application server uses. Requests are sent to the server through any of the HTTP transports. A thread pool enables components of the server to reuse threads to eliminate the need to create new threads at run time. Creating new threads expends time and resources.

To view this administrative console page, click **Servers > Application Servers > *server\_name* > Thread Pools**. (You can reach this page through more than one navigational route.)

*Thread pool settings:*

Use this page to configure a group of threads that an application server uses. Requests are sent to the server through any of the HTTP transport channels or HTTP transports. A thread pool enables components of the server to reuse threads to eliminate the need to create new threads at run time. Creating new threads expends time and resources.

To view this administrative console page, click **Servers > Application Servers > *server\_name* > Thread Pools**, then select the thread pool. (You can reach this page through more than one navigational route.)

*Minimum size:*

Specifies the minimum number of threads to allow in the pool.

<b>Data type</b>	Integer
<b>Default</b>	10

*Maximum size:*

Specifies the maximum number of threads to allow in the pool.

If your Tivoli Performance Viewer shows the Percent Maxed metric to remain consistently in the double digits, consider increasing the Maximum size. The Percent Maxed metric indicates the amount of time that the configured threads are used. If there are several simultaneous clients connecting to the server-side ORB, increase the size to support up to 1000 clients.

<b>Data type</b>	Integer
<b>Default</b>	50
<b>Recommended</b>	50 (25 on Linux systems)

*Thread inactivity timeout:*

Specifies the number of milliseconds of inactivity that should elapse before a thread is reclaimed. A value of 0 indicates not to wait and a negative value (less than 0) means to wait forever.

**Note:** The administrative console does not allow you to set the inactivity timeout to a negative number. To do this you must modify the value directly in the *server.xml* file.

<b>Data type</b>	Integer
<b>Units</b>	Milliseconds
<b>Default</b>	3500

*Allow thread allocation beyond maximum thread size:*

Specifies whether the number of threads can increase beyond the maximum size configured for the thread pool.

<b>Data type</b>	Boolean
<b>Default</b>	Not enabled (false)

## 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 WebSphere Application Server administrative domain, although it is not a server that is supplied by the WebSphere Application Server 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 > 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. WebSphere Application Server 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 WebSphere 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.

<b>Data type</b>	String
<b>Default</b>	

## Starting servers

Starting a server starts a new server process based on the process definition settings of the current server configuration. The node agent for the node on which the Application Server resides must be running before you can start the Application Server.

If you need to restart a server, follow the directions in this article for starting servers. The procedure that applies to starting servers also applies to restarting servers.

**Note:** If you created a new server definition using a base WebSphere Application Server, you cannot start, stop, or manage the new server using the original base Application Server.

**Note:** After you start an Application Server, other processes might not discover the running Application Server immediately. Application Servers are discovered by the node agent. Node agents are discovered by the deployment manager. Node agents usually can discover local Application Servers quickly but it can take a deployment manager from 2 to 60 seconds to discover a node agent.

**Note:** If you are using clusters, note that the **Initial State** property of the Application Server subcomponent (**Servers > Application Server > server\_name > Administration > Server Components > Application Server**) is not intended to be used to control the state of individual servers in the cluster at the time the cluster is started. It is intended only as a way to control the state of the Application Server subcomponent of a server. It is best to start and stop the individual servers of a cluster using the Server options of the Administrative Console or command line commands (**startServer** and **stopServer**).

There are several options for starting an Application Server:

- **Windows** If you are using Windows, you can use the **Start** menu to start the Application Server. If you are using the Network Deployment version of the product, click **Start > Programs > IBM WebSphere > Network Deployment v6.0 > Start the Manager**. You can check that the server has successfully started by checking the `startServer.log` file. If the server has successfully started, the last two lines of the `startServer.log` file reads:

```
Server launched. Waiting for initialization status.  
Server server1 open for e-business; process id is 1932.
```

The `startServer.log` file is located in the `<drive>:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name\logs\server1` directory if you have installed your server with the default settings. The server name and process id vary depending on your settings.

- On distributed platforms, use the **startServer** command to start an Application Server from the command line.

If the node agent for the node on which the Application Server resides is not running, run the **startNode** command and then run the **startServer** command.

- On AIX, you can use the command line to start the server. Use the **startServer** command from the `/usr/WebSphere/AppServer/bin` directory, as shown below. To start a server that is associated with a non-default profile, issue the **startServer** command from the `/opt/WebSphere/AppServer/profiles/profile_name/bin` directory.

```
# ./startServer.sh server1
```

Use the **startManager** command from the `/usr/IBM/WebSphere/AppServer/product/bin` directory:

```
# ./startManager.sh
```



You can check that the server has successfully started by checking the `startServer.log` file. If the server has successfully started, the last two lines of the `startServer.log` file reads:

```
Server launched. Waiting for initialization status.  
Server server1 open for e-business; process id is 1932.
```

On AIX, the `startServer.log` file is located in the `/usr/IBM/WebSphere/AppServer/profiles/profile_name/logs/server1/` directory.

- Start an Application Server using the administrative console.
  1. Click **Servers > Application Servers** from the administrative console navigation tree to go to the Application Server page.
  2. If the node agent for the node on which the Application Server resides is not running, click **Restart** or **Restart all Servers on Node** on the Node Agents page (**System Administration > Node agents**) to start the node agent.

If the node agent does not start, run the **startNode** command and then run the **startServer** command.. Once a node agent completely stops running and remains stopped, you cannot remotely start the node agent from the Node Agents page. You must run the **startNode** command to start the node agent on the node where it runs.
  3. Select the Application Server and click **Start**.
  4. View the **Status** value and any messages or logs to see whether the Application Server starts.
- Start an Application Server for tracing and debugging.

To start the Application Server with standard Java debugging enabled:

  1. Click **Servers > Application Servers** from the administrative console navigation tree. Then, click the Application Server whose processes you want to trace and debug, then **Java and Process Management > Process Definition > Java Virtual Machine**.
  2. On the Java Virtual Machine page, place a checkmark in the check box for the **Debug Mode** setting to enable the standard Java debugger. If needed, set debug arguments. Then, click **OK**.
  3. Save the changes to a configuration file.
  4. Stop the Application Server.
  5. Start the Application Server again as described previously.

Once the server is started, you can install your applications.

## Running application servers from a non-root user

By default, each base WebSphere Application Server node uses the root user ID to run all application server processes. However, you can run all application server processes under the same non-root user and user group. This task describes how to run an application server process from a non-root user.

If global security is enabled, the user registry must not be Local OS. Using the Local OS user registry requires the application server to run as root. Refer to the *Securing applications and their environment* PDF for details.

Run your application servers as non-root when you no longer want to use root authority. For security or administrative reasons, you may want to change to non-root user IDs. Perform this task at any time to change the permissions of an application server. You must restart the application server in order for the changes to take effect.

If your application server is part of a cell, see “Running an application server from a non-root user and the node agent from root” on page 206 or “Running an Application Server and node agent from a non-root user” on page 207

**Note:** If you are using the Tivoli Access Manager (TAM) to perform authentication or authorization for WebSphere Application Server, it is important to be aware of potential permissions problems. For more information, see the *Securing applications and their environment* PDF.

For the following steps, assume that:

- was1 is the user to run the application server
- wasgroup is the primary user group for user was1
- wasnode is the node name
- server1 is the application server
- /opt/WebSphere/AppServer is the installation root
- nodeProfile1 is the profile name.

**Note:** For information about creating a profile, see the *Administering applications and their environment* PDF.

To configure an application server to run as non-root, complete the following steps.

1. Log on to the application server system as the root user.
2. Create the user ID was1 with a primary user group of wasgroup. The user ID, was1, is an example. You can name the user something else.
3. Log off and back on as root.
4. Start server1 as root. Run the startServer.sh script from the /bin directory of the installation root:  
startServer.sh server1
5. Specify user and group ID values for the **Run As User** and **Run As Group** settings for a server:
  - a. Start the administrative console.
  - b. Go to the Process execution page of the administrative console. You must define all three properties in the following table. Click **Servers > Application Servers > server1 > Server Infrastructure > Java and Process Management > Process Execution** and change all of the following values:

Property	Value
Run As User	was1
Run As Group	wasgroup
UMASK	022

- c. Click **OK**.
  - d. Save the configuration.
6. Stop the application server. Use the stopServer.sh script from the /bin directory of the installation root:  
stopServer.sh server1
  7. Change file permissions as the root user. The following example assumes that the installation root directory for WebSphere Application Server is /opt/WebSphere/AppServer:  
chgrp wasgroup /opt/WebSphere  
chgrp wasgroup /opt/WebSphere/AppServer  
chgrp -R wasgroup /opt/WebSphere/AppServer/cloudscape  
chgrp -R wasgroup /opt/WebSphere/AppServer/profiles/nodeProfile1  
chmod g+wr /opt/WebSphere  
chmod g+wr /opt/WebSphere/AppServer  
chmod -R g+wr /opt/WebSphere/AppServer/cloudscape  
chmod -R g+wr /opt/WebSphere/AppServer/profiles/nodeProfile1
  8. Log on to the application server system as was1.
  9. Start server1 as was1. Run the startServer.sh script from the /bin directory of the installation root:  
startServer.sh server1

10. If creating another server with a different user ID, follow this procedure again for the new user ID and server name.

The two user IDs must share the same group, wasgroup.

You can start an application server from a non-root user.

## Running an application server from a non-root user and the node agent from root

By default, each base WebSphere Application Server node on a Linux and UNIX platform uses the root user to run application servers. However, you can use a non-root user to run application servers. This task describes how to configure an application server to run as non-root while letting the node agent process run as root.

If global security is enabled, it is not recommended that the Local OS be used for user registry. In general, using the Local OS user registry requires that all processes run as root. Refer to the *Securing applications and their environment* PDF for details. If you are attempting to run an Application Server as root in WebSphere Application Server Version 6 when you previously used a non-root user ID on Linux and UNIX platforms in Version 5.x, see the *Migrating, coexisting, and interoperating* PDF.

Using a non-root user ID to run application servers can be done by setting all the application servers to run under the same operating system group. Run your application servers as non-root when you no longer want to use root authority. For security or administrative reasons, you may want to change to non-root user IDs. Perform this task at any time to change the permissions of an application server. You must restart the application servers in order for the changes to take effect.

**Note:** If you are using the Tivoli Access Manager (TAM) to perform authentication or authorization for WebSphere Application Server, it is important to be aware of potential permissions problems. For more information, see the *Securing applications and their environment* PDF.

1. Log on to the application server system as root.
2. Create the was1 user and wasgroup group that you can use to run the application server. If you will be using peer recovery with your transaction logs on a shared system (such as NAS), between two or more machines, create users and groups with the same identification numbers on all machines participating in peer recovery. This ensures that the non-root users and groups match across machines.
3. Add users root and was1 to the wasgroup group.
4. Log off and back on.
5. Log on to the Network Deployment system as root.
6. If it is not started, start the deployment manager process with the startManager.sh script from the /bin directory of the installation root:
 

```
startManager.sh
```
7. Configure application server properties for the root and was1 users. Use the administrative console on the deployment manager to complete the following steps:
  - a. Define the node agent to run as a root process. You must define all three properties in the following table. Click **System Administration > Node agents > nodeagent (for the node)Server Infrastructure > Java and Process Management > Process Definition > Process Execution** and change all of the following values:

Property	Value
Run As User	root
Run As Group	wasgroup
UMASK	002

- b. Define each application server to run as a was1 process. Substitute the name of each server for server1. You must define all three properties in the following table. Click **Servers > Application Servers > server1 > Server Infrastructure > Java and Process Management > Process Definition > Process Execution** and change all of the following values:

Property	Value
Run As User	was1
Run As Group	wasgroup
UMASK	002

- c. Save and synchronize all nodes.
- Log on to the application server system as root.
  - Ensure that all servers on the application server system are stopped, including the server1 process. Use the stopServer.sh script from the /bin directory of the installation root:  

```
stopServer.sh server1 -user userID -password password
```
  - Ensure that the node agent process is stopped. Use the stopNode.sh script from the /bin directory of the installation root:  

```
stopNode.sh -user userID -password password
```
  - As root, use operating system tools to change the following file permissions on the application server system:  

```
chgrp wasgroup /opt/WebSphere
chgrp wasgroup /opt/WebSphere/AppServer
chgrp -R wasgroup /opt/WebSphere/AppServer/cloudscape
chgrp -R wasgroup /opt/WebSphere/AppServer/profiles/nodeProfile1
chmod g+wr /opt/WebSphere
chmod g+wr /opt/WebSphere/AppServer
chmod -R g+wr /opt/WebSphere/AppServer/cloudscape
chmod -R g+wr /opt/WebSphere/AppServer/profiles/nodeProfile1
```
  - Start the node agent process from root. Use the startNode.sh script from the /bin directory of the installation root:  

```
startNode.sh
```
  - Log on to the application server system as the was1 user.
  - Start all application servers from the was1 user. Use the startServer.sh script from the /bin directory of the installation root:  

```
startServer.sh server1
```

You can start an application server from a non-root user and run the node agent as root.

## Running an Application Server and node agent from a non-root user

By default, each base Application Server node on a Linux, or UNIX operating system uses the root user ID to run the node agent process and all Application Server processes. However, you can run the node agent and all Application Server processes under the same non-root user and user group. If you do run the node agent process with a non-root user ID, you must run all Application Server processes that the node agent controls under the same non-root user ID.

If global security is enabled, the user registry must not be Local OS. Using the Local OS user registry requires the node agent to run as root. Refer to the *Securing applications and their environment* PDF for details.

Using the same non-root user and user group gives the node agent process the operating system permissions to start all other server processes.

Run your application servers and node agent as non-root when you no longer want to use root authority. For security or administrative reasons, you may want to change to non-root user IDs. Perform this task at any time to change the permissions of a node agent or application server. You must restart the node agent and application servers in order for the changes to take effect.

**Note:** If you are using the Tivoli Access Manager (TAM) to perform authentication or authorization for WebSphere Application Server, it is important to be aware of potential permissions problems. For more information, see the *Securing applications and their environment* PDF.

For the steps that follow, assume that:

- wasadmin is the user to run all servers
- wasnode is the node name
- wascell is the cell name
- server1 is the Application Server
- /opt/WebSphere/AppServer is the installation root for the base node
- wasgroup is the group that will run all servers, with wasadmin as a member
- nodeProfile1 is the profile name

**Note:** For information about creating a profile, see the *Administering applications and their environment* PDF.

To configure a user ID to run the node agent and all server processes, complete the following steps.

1. Log on to the Application Server system as root.
2. Create user wasadmin with primary group wasgroup. If you will be using peer recovery with your transaction logs on a shared system (such as NAS) between two or more machines, you will need to create a user and group with the same identification numbers on all machines participating in peer recovery. This will ensure that the non-root users and groups match across machines.
3. Log off and back on.
4. Log on to the Network Deployment system as root.
5. If the deployment manager process is not started, start it with the startManager.sh script from the /bin directory of the installation root:  
startManager.sh
6. Start the administrative console.
7. Define the node agent to run as a wasadmin process using the administrative console of the deployment manager. You must define all three properties in the following table. Click **System Administration > Node agents > nodeagent (for the node) Server Infrastructure > Java and Process Management > Process Definition > Process Execution** and change all of the following values:

Property	Value
Run As User	wasadmin
Run As Group	wasgroup
UMASK	022

**Note:** Make sure that the node agent is running if you are going to change the value specified for either the Run As Group or Run As User property. If the value for either of these properties is changed while the node agent is not running, the Deployment Manager can not push the changes to the node.

8. Define each Application Server to run as a wasadmin process. Substitute the name of each server for server1. You must define all three properties in the following table. Click **Servers > Application Servers > server1 > Server Infrastructure > Java and Process Management > Process Execution** and change all of the following values:

Property	Value
Run As User	wasadmin
Run As Group	wasgroup
UMASK	022

9. Save and synchronize all nodes. Stop all changed application servers and the node agent from the administrative console.
10. Log on to the Application Server system as root.
11. Ensure that all servers and the node agent are stopped.
12. As root, use operating system tools to change file permissions on Linux and UNIX platforms:

```

chgrp wasgroup /opt/WebSphere
chgrp wasgroup /opt/WebSphere/AppServer
chgrp -R wasgroup /opt/WebSphere/AppServer/cloudscape
chgrp -R wasgroup /opt/WebSphere/AppServer/profiles/nodeProfile1
chmod g+wr /opt/WebSphere
chmod g+wr /opt/WebSphere/AppServer
chmod -R g+wr /opt/WebSphere/AppServer/cloudscape
chmod -R g+wr /opt/WebSphere/AppServer/profiles/nodeProfile1

```
13. Log in as wasadmin on the Application Server system.
14. From wasadmin, run the startNode.sh script from the /bin directory of the installation root to start the node agent:

```

startnode.sh node1

```
15. Log into the administrative console and start the application servers.

You can start an application server and the node agent from a non-root user.

## Detecting and handling problems with run-time components

You must monitor the status of run-time components to ensure that, once started, they remain operational as needed.

1. Regularly examine the status of run-time components. Browse messages displayed under **Websphere Runtime Messages** in the status area at the bottom of the console. The run-time event messages marked with a red **X** provide detailed information on event processing. You can also use the Logging and Tracing page of the administrative console to monitor the status of run-time components. Click **Troubleshooting > Logs and Trace** in the console navigation tree to access the page.
2. If an application stops running when it should be operational, examine the application's status 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 has stopped running, use the Server Cluster page to try restarting 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 run-time components do not restart, reexamine the messages and read information on problem determination to help you to restart the components.

## Stopping servers

Stopping an application server stops a server process based on the process definition settings in the current application server configuration.

- **Windows** In Windows, you can use the Start menu to stop your application server. Click **Start > Programs > IBM WebSphere > Network Deployment v6.0 > Stop the server**. When the server stops successfully, the stopServer.log file contains the following in the last two lines:



Server stop request issued. Waiting for stop status.  
Server server1 stop completed.

The server name varies depending on your settings.

- Stop the application server from the command line. In distributed environments, you can use the **stopServer** command to stop a single server or the **stopManager** command to stop the deployment manager. In AIX, use the **stopServer** or the **stopManager** command from the `/usr/WebSphere/AppServer/bin` directory:

```
# ./stopServer.sh server1  
# ./stopManager.sh
```

- Use the administrative console to stop an application server:
  1. In the administrative console, click **Servers > 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.

## 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 > Application servers > server**. Then under **Additional properties**, select **Core group service**.

Click **Save** to save and synchronize your changes with all managed nodes.

### 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

When selected, 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.

**Important:** 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.



<b>Data type</b>	String
<b>Default</b>	10 megabytes

---

## Creating generic servers

There are two types of generic application servers:

- Non-Java applications or processes.
- Java applications or processes

You can use the wsadmin tool or the **Generic servers** panel of the administrative console to create either type.

**Note:** For the Base WebSphere Application Server product, although you can use the administrative console to create a generic application server definition, you cannot use it to start, stop or, in any way, control or manage that application server. The Base product administrative console can only be used to create server definitions and, if necessary, adjust the server definitions that it creates. To manage Base generic application servers, use the wsadmin tool.

- **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**. You can then specify the name of the generic server you are creating.
  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 Servers. This will enable you to quickly determine whether to use the **Terminate** or **Stop** button in the administrative console to stop specific application server. You must use the **Terminate** button to stop a generic application server.
  4. Select a template to use in creating the new server. You can use a default application server template for your new server or use an existing application server as a template. The new application server will inherit all properties of the template server. If you create the new server using an existing application server do not enable the option to map applications from the existing server to the new server. This option does not apply for a generic server.
  5. Click **Next**
  6. Click **Finish**. The generic server now appears as an option on the **Generic servers** panel in the administrative console.
  7. On the **Generic servers** panel, 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 name of the non-WebSphere Application Server program that is to be launched when you start this generic server. 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
  10. 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 > Generic servers**
  2. Click **New**. You can then specify the name of the generic server you are creating.
  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 Servers. This will enable you to quickly determine whether to use the **Terminate** or **Stop** button in the administrative console to stop specific application server. You must use the **Terminate** button to stop a generic application server.

4. Click **Next**
5. Click **Finish**. The generic server now appears as an option on the **Applications Server** panel in the administrative console.
6. Click **Finish**. The generic server now appears as an option on the **Generic servers** panel in the administrative console.
7. On the **Generic servers** panel, 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 Websphere Application Server's default JVM (`{JAVA_HOME}/bin/java`), which will be 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**, will be used as the executable target of this Java process. The default for Websphere Application Server 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 will be either a Java class containing a `main()` method, or the name of an executable JAR file.) The default for Websphere Application Server is **com.ibm.ws.runtime.WsServer**.
12. Click **OK**.

**Note:** If the generic server is to run an application server other than the WebSphere Application 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.

You can now start and terminate the generic server whenever you want to start or terminate the non-WebSphere Application Server server or process associated with this server.

## Starting and terminating generic servers

This topic describes how to start and terminate generic servers.

If you created a generic server on a Base WebSphere Application Server, you cannot start, terminate, or monitor this server with the Base Application Server administrative console. You must use the `wsadmin` tool to manage Base generic servers.

### Starting generic servers

There are two ways to start a generic server in a Network Deployment environment:

- Use the administrative console:
  1. From the administrative console navigation tree, select **Servers > Application Servers**.
  2. Select the check box beside the name of the generic server, and then click **Start**.
  3. View the **Status** value and any messages or logs to see whether the generic server starts.
- Use the MBean `NodeAgent launchProcess` operation of the `wsadmin` tool.

### Terminating generic servers

There are two ways to terminate a generic server in a Network Deployment environment:

- Use the administrative console:
  1. From the administrative console navigation tree, select **Servers > Application Servers**.
  2. Select the check box beside the name of the generic server, and then click **Terminate**.
  3. View the **Status** value and any messages or logs to see whether the generic server terminates.

**Note:** The **Stop** and **Stop Immediate** buttons on the administrative console do not work for generic servers.

- Use the MBean terminate launchProcess operation of the wsadmin tool.

---

## Configuring transport chains

You need to configure transport chains to provide networking services to such functions as 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 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. To define these channels:

1. Create a transport chain: You can either use the administrative console or wsadmin commands to create a transport chain. If you want to use the administrative console:
  - a. Ensure that a port is available for the new transport chain.
  - b. In the administrative console, click **Servers > Application servers > server\_name**, and then click on one of the following:
    - Under **Web container settings**, click **Web container transport chains**.
    - Under **Server messaging**, click either **Messaging engine inbound transports** or **WebSphere MQ link inbound transports**.
  - c. 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 will be 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.

When you click **Finish**, the new transport chain is added to the list of defined transport chains on the **Transport chain** panel.

2. Click on the transport chain's name to view the configuration settings that are in effect for the transport channels contained in this chain. To change any of these settings:
  - a. Click on the channel that requires changes to its settings.
  - b. Make your changes to 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 have been defined for your system.
3. When you have made all of your changes, click **OK**.
4. Stop the application server and start it again. You must stop the application server and start it again before the configuration changes you made take affect.

## 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.

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.

#### **TCP channel**

Used to provide client applications with persistent connections within a Local Area Network (LAN). When configuring a TCP channel, you can specify a list of IP addresses that are allowed to make inbound connections and a list of IP addresses that are not allowed to make inbound connections. You can also specify the thread pool that this channel uses, which allows you to segregate work by the port that the application server is listening on.

#### **HTTP 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 server HTTP requests and to send HTTP specific information to servlets expecting this type of information.

#### **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.

#### **Web container channel**

Used to create a bridge in the transport chain between an HTTP inbound channel and a servlet and JavaServer Pages (JSP) engine.

#### **DCS channel**

Used 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.

#### **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.

#### **JFAP channel**

Used by the Java Message Service (JMS) server to create connections to JMS resources on a service integration bus.

#### **SSL channel**

Used to associate an SSL configuration repertoire with the transport chain. This channel is only available when Secure Sockets Layer (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.

### **HTTP transport channel custom property**

If you are using an HTTP transport channel, you can add the following custom property to the configuration settings for that HTTP transport channel.

To add a custom property:

1. In the administrative console, click **Application servers** > *server\_name* **Web container settings** > **Web container transport chains** > *chain\_name* > **HTTP Inbound Channel** > **Custom Properties** > **New**
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 is a list of custom properties provided with the application server. These properties are not shown on the settings page for an HTTP transport channel.

### **inProcessLogFilenamePrefix**

Use 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`.

**Data type** String

## **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 HTTP Tunnel transport channel.

To add a custom property:

1. In the administrative console, click **Servers** > **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 custom property that is provided with the application server. 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

## Troubleshooting transport chain problems

### 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 a WebSphere 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 window to display the state of the port to which you are trying to bind. If you need to change the amount of elapse time that must occur before TCP/IP can release a closed connection and reuse its resources, see the *Troubleshooting and support* PDF.

## Configuring HTTP transports

**Important:** On a distributed platform, HTTP transport support is deprecated. Therefore, the administrative console page used to configure an HTTP transport is not available unless you migrated an HTTP transport from your V5 environment. You must define an HTTP transport channel instead of an HTTP transport to handle your HTTP requests.

An HTTP transport is the request queue between a WebSphere Application Server plug-in for Web servers and a Web container in which the Web modules of an application reside. To define the characteristics of the connections between that plug-in and the Web container, you must specify:

- How the transport is to handle a set of connections. For example, you must specify the number of concurrent requests that is to be allowed.
- Whether to secure the connections with SSL.
- The Host and IP information for the transport participants.

1. Change the configuration for an existing HTTP transport.
  - a. Ensure that virtual host aliases include port values for the transport you are changing.
  - b. Go to the HTTP Transports page and click on the transport under **Host** whose configuration you want to change.

Remember, on a distributed platform, HTTP transport support is deprecated. Therefore you cannot view this administrative console page unless you are a V5.x user who, during the V6 migration process, indicated that you want to continue using the HTTP transports that are defined for your V5 environment.

- c. On the settings page for an HTTP transport, which might have the page title `DefaultSSLSettings`, change the specified values as needed, then click **OK**.
  - d. Custom properties page, add and set any custom properties you want to use.
2. Stop the WebSphere Application Server and start it again. You must stop the WebSphere Application Server and start it again before the configuration changes you made take affect.

If the Web server is located on a machine remote from the Application Server but is defined on a managed node, the updated `plugin-cfg.xml` is automatically propagated to the Web server.

If the Web server is defined on an unmanaged node and is not an IHS V6.0 Web server, copy the updated `plugin-cfg.xml` file to the remote Web server and replace the file that is there. See the *Installing your application serving environment* PDF for information about copying the `plugin-cfg.xml` and binary plug-in module to a remote Web server and configuring the Web server to use the files.



If the Web server is an IHS V6.0 Web server, is located on a machine remote from the Application Server and is defined on an unmanaged node, you can configure the `plugin-cfg.xml` file so that whenever it is updated, it is automatically propagated to the remote IHS Web server.

## HTTP transport collection

Use this page to view or manage HTTP transports. Transports provide request queues between WebSphere Application Server plug-ins for Web servers 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.

On a distributed platform, if, when migrating from WebSphere Application Server Version 5.x, you indicate that you want to continue using an HTTP transport to handle your HTTP requests, your Version 5.x transports are migrated for you. If you are not migrating from Version 5.x, you must set up an HTTP transport channel to handle your HTTP requests.

To view the HTTP Transport administrative console page, click **Servers > Application Servers > *server\_name* > Web Container Settings > Web Container > HTTP Transports.**

the HTTP Transport panel on the administrative console

### **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 distributed platforms, there is no limit to the number of HTTP ports that are allowed per process.

### **SSL Enabled:**

Specifies whether to protect connections between the WebSphere plug-in and application server 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`.

On a distributed platform, if, when migrating from Version 5, you indicate that you want to continue using an HTTP transport to handle your HTTP requests, your Version 5 transports are migrated for you. If you are not migrating from WebSphere Application Server Version 5.x, you must set up an HTTP transport channel to handle your HTTP requests.

To view the HTTP Transport panel on the administrative console, click **Servers > Application Servers > *server\_name* > Web Container Settings > Web Container > HTTP Transports > *host\_name*.**

### **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`.

<b>Data type</b>	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.

<b>Data type</b>	Integer
<b>Range</b>	1 to 65535

### **SSL Enabled:**

Specifies whether to protect connections between the WebSphere Application Server plug-in and application server with Secure Sockets Layer (SSL). The default is not to use SSL.

<b>Data type</b>	Boolean
<b>Default</b>	false

### **SSL:**

Specifies the Secure Sockets Layer (SSL) settings type for connections between the WebSphere Application Server plug-in and application server. The options include one or more SSL settings defined in the Security Center; for example, DefaultSSLSettings, ORBSSLSettings, or LDAPSSLSettings.

<b>Data type</b>	String
<b>Default</b>	An SSL setting defined in the Security Center

## **Transports**

A transport is the request queue between a WebSphere Application Server plug-in for Web servers and a Web container in which the Web modules of an application reside. When a user at a Web browser requests an application, the request is passed to the Web server, then along the transport to the Web container.

Transports define the characteristics of the connections between a Web server and an application server, across which requests for applications are routed. Specifically, they define the connection between the Web server plug-in and the Web container of the application server.

Administering transports is closely related to administering WebSphere Application Server plug-ins for Web servers. Indeed, without a plug-in configuration, a transport configuration is of little use.

On a distributed platform, when migrating from WebSphere Application Server Version 5.x, you indicate that you want to continue using an HTTP transport to handle your HTTP requests, your Version 5.x transports are migrated for you. If you are not migrating from Version 5.x, you must set up an HTTP transport channel to handle your HTTP requests.

### **The internal transport**

The internal HTTP transport allows HTTP requests to be routed to the application server directly through a Web server plug-in. Logging is provided for debug purposes.

Prior to WebSphere Application Server Version 5.0.2, the HTTP transport functionality existed only as a means of accepting HTTP requests forwarded by an HTTP plug-in that was connected to a Web server. In WebSphere Application Server Version 5.0.2, HTTP transport functionality is now a supported internal Web server. By default, the internal HTTP transport listens for HTTP requests on port 9080 and for HTTPS requests on port 9443.

For example, use the URL `http://localhost:9080/snoop` to send requests to the snoop servlet on the local machine over HTTP and `https://localhost:9443/snoop` to send requests to the snoop servlet on the local machine over HTTPS.

The transport configuration is a part of the Web container configuration. You can configure the internal transport to use ports other than 9080 and 9443. However, you must also adjust your virtual host alias and what you type into the Web browser.

## HTTP transport custom properties

Use this page to set custom properties for an HTTP transport.

On a distributed platform, HTTP transport support is deprecated. Therefore you cannot create a new HTTP transport. Instead you must create an HTTP transport channel to handle your HTTP requests. If you are a WebSphere Application Server Version 5.x user who has migrated to Version 6, and during the migration process you indicated that you want to continue using an HTTP transport to handle your HTTP requests, your Version 5.x transports are still available for your use.

If you are using HTTP transports, you can set the following custom properties on either the Web Container or HTTP Transport **Custom Properties** panel on 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 values for these custom properties for a specific transport on the HTTP Transport **Custom Properties** page:

1. In the console navigation tree, click **Servers > Application Servers > server\_name > Web Container settings > Web Container > HTTP Transport**

To specify a custom property:

1. Click on the **HOST** whose properties you want to set.
2. Under **Additional Properties** select **Custom Properties**.
3. On the Custom Properties page, click **New**.
4. 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.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. 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 an HTTP transport.

### ***ConnectionIOTimeout:***

Use the `ConnectionIOTimeout` property to specify the maximum number of seconds to wait when trying to read or process data during a request.

<b>Data type</b>	Integer
<b>Default</b>	For distributed platforms: 5 seconds

### ***ConnectionKeepAliveTimeout:***

Use the `ConnectionKeepAliveTimeout` property to specify the maximum number of seconds to wait for the next request on a keep alive connection.

<b>Data type</b>	Integer
------------------	---------

### ***MaxConnectBacklog:***

Use the `MaxConnectBacklog` property to specify the maximum number of outstanding connect requests that the operating system will 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 will be rejected.

Set this value to the number of concurrent connections that you would like to allow. Keep in mind that a single client browser might need to open multiple concurrent connections (perhaps 4 or 5); however, also keep in mind that increasing this value consumes more kernel resources. The value of this property is specific to each transport.

<b>Data type</b>	Integer
<b>Default</b>	511

### ***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.

<b>Data type</b>	Integer
<b>Default</b>	For distributed platforms: 100 requests

### ***KeepAliveEnabled:***

This property is only valid in a distributed environment. Use the `KeepAliveEnabled` property to specify whether or not to keep connections alive

<b>Data type</b>	String
<b>Default</b>	true

### ***Trusted:***

This property is only valid in a distributed environment. Use the `Trusted` property to indicate that the application server can use the private headers that the Web server plug-in adds to requests.

<b>Data type</b>	String
<b>Default</b>	false

## **Configuring error logging for internal Web server HTTP transport**

To debug potential problems with using the HTTP transport as an internal Web server, you can use the following error logging capabilities.

1. Turn error logging on. To turn error logging on, add the following custom property to the HTTP Transport's configuration settings, and set the value to **false**:

```
Property name: ErrorLogDisable  
Value: True/False  
Default: Error log is disabled by default
```

When you are ready to turn error logging off, set the value of the **ErrorLogDisable** property back to **true**.

2. To specify your own error log file, add the following property to the transport section of the server.xml file:

Property name: ErrorLog  
Value: <filename>  
Default: logs/<server instance>/http.log

The error log property is used to specify where to place the error log. For example:<properties xmi:id="WebContainer\_Property\_6" name="ErrorLog" value="logs/<server instance>/http.log"/>

**Note:** The error log should appear in each instance of the server.

If you are going to be using error logging for multiple HTTP transports in a single HTTP server, make sure you specify a unique filename for the error log file associated with each HTTP transport.

3. Add the LogLevel property to the transport section of the server.xml file to specify the level of messages to log in the error log file.

Property name: LogLevel  
Value: <level> (Levels include: debug, info, warn, error, crit)  
Default: warn (warn includes error and crit; debug includes all levels)  
Scope: Virtual/Global

Log levels specify the type of message that appears in the error log. The *warn*, *error*, and *crit* messages are logged by default.

4. Restart the server.

If you have enabled error logging and encounter an error, there should be an error log message in the error log file you specified.

## Configuring access logging for internal Web server HTTP transport

To debug potential problems with using the HTTP transport as an internal Web server, you can use the following access logging capabilities.

1. Turn access logging on. To turn access logging on, add the following custom property to the HTTP Transport's configuration settings, and set the value to **false**:

Property name: AccessLogDisable  
Values: True/False  
Default: Access log is disabled by default

When you are ready to turn access logging off, set the value of the **AccessLogDisable** property back to **true**.

2. To specify your own access log file, add the following property to the transport section of the server.xml file:

Property name: AccessLog  
Value: <filename>  
Default Value: logs/<server instance>/http\_access.log

The default access log file is logs/<server\_instance>/http\_access.log. Access log entries should have the format:

<hostname or IP> <user agent> [<local time> -<status code>] <thread id> <http request> <status code> <bytecount>

**Note:** If you are going to be using access logging for multiple HTTP transports in a single HTTP server, make sure you specify a unique filename for the access log file associated with each HTTP transport.

3. Restart the server.

If you have enabled access logging, there will be an access log in the location you specified.

## Transport chains collection

Use this page to view or manage transport chains. Transport chains enable communication through transports, 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 > 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. For WebSphere Application Server, transport name must be unique within a WebSphere Application Server configuration. Click on the name of a transport chain to change its configuration settings.

### Enabled

When set to true, the transport chain is activated at application server startup.

### 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, 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 set to true, users are notified if there is a channel that enables Secure Sockets Layer (SSL) in the listed chain. When SSL is enabled, all traffic going through this transport is encrypted and digitally secured.

## Transport chain settings

This page lists 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 > Application servers > *server\_name* > Ports**. Click on **View associated transports** for the port whose transport chains you want view and then click on 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 a WebSphere Application Server configuration.

### Enabled

When checked, this transport chain is activated at application server startup.

## Transport channels

Lists the transport channels configured for this transport chain and their configuration settings. To change a transport channel's configuration settings, click on the name of that transport 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 > 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.

This name must be unique across all channels in a WebSphere Application Server environment. For example DCS and TCP transport channels cannot have the same name if they reside within the same 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.

<b>Data type</b>	Positive integer
<b>Default</b>	10

## 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 > Application servers > *server\_name* > Ports** . Click on **View associated transports** for the port associated with the HTTP transport channel whose settings you want to look at.

### Transport channel name

Specifies the name of the HTTP transport channel.

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.

### 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.

<b>Data type</b>	Positive integer
<b>Default</b>	10

### Maximum persistent requests

Specifies the maximum number of persistent (keep-alive) requests that are allowed on a single HTTP connection. If a value of 0 (zero) is specified, only one request is allowed per connection. If a value of -1 is specified, an unlimited number of requests is allowed per connection.

<b>Data type</b>	Integer
<b>Default</b>	100

### Use Keep-Alive

When selected, the HTTP transport channel, when sending an outgoing HTTP message, uses a persistent connection (keep-alive connection) instead of a connection that closes after one request or response exchange occurs.

**Note:** If a value other than 0 is specified for the maximum persistent requests property, the Use Keep-Alive property setting is ignored.

The default for this property is selected.

### Read timeout

Specifies the amount of time, in seconds, the HTTP transport channel waits for a read request to complete on a socket after the first read request occurs. The read being waited for could be an HTTP body (such as a POST) or part of the headers if they were not all read as part of the first read request on the socket.

<b>Data type</b>	Integer
<b>Default</b>	60 seconds

### Write timeout

Specifies the amount of time, in seconds, that the HTTP transport channel waits on a socket for each portion of response data to be transmitted. This timeout usually only occurs in situations where the writes are lagging behind new requests. This can occur when a client has a low data rate or the server's network interface card (NIC) is saturated with I/O.

<b>Data type</b>	Integer
<b>Default</b>	60 seconds

### Persistent timeout

Specifies the amount of time, in seconds, that the HTTP transport channel allows a socket to remain idle between requests.

<b>Data type</b>	Integer
<b>Default</b>	30 seconds

### Enable NCSA access logging

When selected, the HTTP transport channel performs NCSA access and error logging. Enabling NCSA access and error logging slows server performance.

To configure NCSA access and error logging, click **HTTP error and NCSA access logging** under **Related Items**. Even if HTTP error and NCSA access logging is configured, it is not enabled unless the Enable NCSA access logging property is selected.



The default value for the Enable NCSA access logging property is not selected.

## TCP transport channel settings

Use this page to view and configure an 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 > 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.

This name must be unique across all channels in a WebSphere Application Server environment. For example, a TCP transport channel and an HTTP transport channel cannot have the same name if they reside within the same system.

### 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.

### Thread pool

Select from the drop-down list of available thread pools the thread pool you want the TCP transport channel to use when dispatching work.

### Maximum open connections

Specifies the maximum number of connections that can be open at one time.

<b>Data type</b>	Integer between 1 and 20,000 inclusive
<b>Default</b>	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.

**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.

<b>Data type</b>	Integer
<b>Default</b>	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 **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 can not appear any where 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 can not appear any where 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 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 chains. 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 > Application servers > server\_name > Ports > .** Click on **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.

This name must be unique across all channels in a WebSphere Application Server environment. For example DCS and TCP transport channels cannot have the same name if they reside within the same 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.

The discrimination weight of the DCS channel in a DCS-Secure transport chain should always be less than the discrimination weight of the SSL channel that is in that chain. Other SSL channels in other chains might have different discrimination values.

<b>Data type</b>	Positive integer
<b>Default</b>	1 for the DCS channel                      2 for the SSL channel

## Web container 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 > Application servers > *server\_instance* > Web container settings > Web container transport chains > *transport\_chain* > Web Container Inbound Channel** .

### Transport Channel Name

This name must be unique across all channels in a WebSphere Application Server environment. This means that TCP transport channels and HTTP transport channels cannot have the same name if they reside within the same system.

### Discrimination weight

Specifies the priority that this transport chain has in relation to other transport chains if this transport channel is shared amongst several transport chains.

### 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.

<b>Data type</b>	bytes
<b>Default</b>	9192 bytes

---

## Custom services

A custom service provides the ability to plug into a WebSphere Application Server application server to define a hook point that runs when the server starts and shuts down.

A developer implements a custom service containing a class that implements a particular interface. The administrator configures the custom service in the administrative console, identifying the class created by the developer. When an application server starts, any custom services defined for the application server are loaded and the server runtime calls their initialize methods.

---

## Developing custom services

To define a hook point to be run when a server or node agent starts and shuts down, you develop a custom service class and then configure a custom service instance. When the application server or node agent starts, the custom service starts and initializes.

The following restrictions apply to the WebSphere Application Server custom services implementation:

- The init 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 init and shutdown methods are called only once on each service, and once for each operating system process that makes up the server instance. File I/O is supported.
- 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 J2EE code (client code, servlets, enterprise beans) is not supported.
- The JTA interface is not available. This feature is available in J2EE 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.

Note that these restrictions apply to the shutdown and init methods equally. Some JNDI operations are available.

1. Develop a custom service class that implements the `com.ibm.websphere.runtime.CustomService` interface. The properties passed by the application server runtime to the initialize method can include one for an external file containing configuration information for the service (retrieved with `externalConfigURLKey`). In addition, the properties can contain any name-value pairs that are stored for the service, along with the other system administration configuration data for the service. The properties are passed to the initialize method of the service as a `Properties` object.

There is a shutdown method for the interface as well. Both methods of the interface declare that they may create an exception, although no specific exception subclass is defined. If an exception is created, the runtime logs it, disables the custom service, and proceeds with starting the server.

2. On the Custom Service page of the administrative console, click **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. Doing this adds the path name to the WebSphere Application Server extension classloader, which allows the custom service to locate and correctly load the native library.

3. If you are developing a custom service for an application server, stop the application server and then restart the server.
4. If you are developing a custom service for a node agent, stop and then restart the processing of the node agent. On the Node Agents page of the administrative console (System Administration > Node Agents), place a checkmark in the check box beside the node agent you want to stop, then click Stop. To restart the node agent, place a checkmark in the check box beside the node agent, then click Restart.

5. Check the application server or node agent to ensure that the initialize method of the custom service ran as intended. Also ensure that the shutdown method performs as intended when the server or node agent stops.

As mentioned above, your custom services class must implement the CustomService interface. In addition, your class must implement the initialize and shutdown methods. Suppose the name of the class that implements your custom service is *ServerInit*, your code would declare this class as shown below. The code below assumes that your custom services class needs a configuration file. It shows how to process the input parameter in order to get the configuration file. If your class does not require a configuration file, the code that processes configProperties is not needed.

```
public class ServerInit implements CustomService
{
/**
 * The initialize method is called by the application server run-time when the
 * server starts. The Properties object passed to this method must contain all
 * configuration information necessary for this service 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 run-time when the
 * server begins its shutdown processing.
 *
 * @param configProperties java.util.Properties
 */
    public void shutdown() throws Exception
    {
        // Implement shutdown method
    }
}
```

## 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 a WebSphere application server and define code that runs when the server starts or shuts down.

To view this administrative console page, click **Servers > Application servers >server\_name**. Then, under Server Infrastructure, click **Administration > Custom Services**.

If you are developing a custom service for a node agent, click **System Administration > Node agents >node\_agent\_name**. Then, under Additional Properties, click **Custom Services** to view this administrative console page.

## 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 > Application servers >server\_name**. Then, under Server Infrastructure, click **Administration > Custom services >custom\_service\_name**.

If you are developing a custom service for a node agent, click **System administration > Node agents >node\_agent\_name**. Then, under Additional Properties, click **Custom services >custom\_service\_name** to view this administrative console page.

#### ***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.

<b>Data type</b>	Boolean
<b>Default</b>	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.

<b>Data type</b>	String
<b>Units</b>	URL

#### ***Classname:***

Specifies the class name of the service implementation. This class must implement the Custom Service interface.

<b>Data type</b>	String
<b>Units</b>	Java class name

#### ***Display Name:***

Specifies the name of the service.

<b>Data type</b>	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

---

## Process definition

A process definition specifies the run-time characteristics of an application server process.

A process definition can include characteristics such as JVM settings, standard in, error and output paths, and the user ID and password under which a server runs.

---

## 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 run-time properties such as the program to run, arguments to run the program, and the working directory.

1. Go to the settings page for a process definition in the administrative console. Click **Servers > Application Servers** in the console navigation tree, click on an application server name and then **Java and Process Management > Process Definition**. Note that you can also define application server processes using the wsadmin tool. For more information, see the *Administering applications and their environment* PDF.
2. 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**.
3. Specify process execution statements for starting or initializing a UNIX process.
4. Specify monitoring policies to track the performance of a process.
5. 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.
6. Specify name-value pairs for properties needed by the process definition.
7. Stop the application server and then restart the server.
8. Check the application server to ensure that the process definition runs and operates as intended.

## Process definition settings

Use this page to view or change settings for a process definition. For WebSphere Application Server, this page provides command-line information for starting or initializing a process.

To view this administrative console page, click **Servers > Application Servers > server\_name**. Then under Server Infrastructure click **Java Process Management > Process Definition**.

## Working Directory

Specifies the file system directory that the process uses as its current working directory.

The process uses this directory to determine the locations of input and output files with relative path names.

Passivated enterprise beans are placed in the current working directory of the application server on which the beans are running. Make sure the working directory is a known directory under the root directory of the WebSphere Application Server product.

**Data type** String

## Process execution settings

Use this page to view or change the process execution settings for a server process that applies to either an application server, a node agent or a deployment manager.

To view this administrative console page for an application server, click **Servers > Application Servers > *server\_name***. Then, under Server Infrastructure , click **Java and Process Management > Process Execution**.

To view this administrative console page for a node agent, click **System Administration > Node agents > *node\_agent\_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, under Server Infrastructure , click **Java and Process Management > Process Definition > 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 for WebSphere Application Server on all operating systems.

### ***UMASK:***

Specifies the user mask under which the process runs (the file-mode permission mask).

**Data type** Integer

### ***Run As User:***

Specifies the user that the process runs as.

**Data type** String

### ***Run As Group:***

Specifies the group that the process is a member of and runs as.

On OS/400, the Run As Group setting is ignored.

**Data type** String

### ***Run In Process Group:***

Specifies a specific process group for the process. This process group is useful for such things as processor partitioning. A system administrator can assign a process group to run on, for example, 6 of 12 processors. The default (0) is not to assign the process to any specific group.

On OS/400, the Run In Process Group setting is ignored.

<b>Data type</b>	Integer
<b>Default</b>	0

## **Process logs settings**

Use this page to view or change settings for specifying the files to which standard out and standard error streams write.

To view this administrative console page, click **Servers > Application Servers >server\_name**. Then, under Troubleshooting, click **Logging and Tracing > Process Logs**.

### ***Stdout File Name:***

Specifies the file to which the standard output stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the Runtime tab to select a file for viewing. View the file by clicking **View**.

Direct server output to the administrative console or to the process that launched the server, by either deleting the file name or specifying console on the configuration tab.

<b>Data type</b>	String
<b>Units</b>	File path name

### ***Stderr File Name:***

Specifies the file to which the standard error stream is directed. The file name can include a symbolic path name defined in the variable entries.

Use the field on the configuration tab to specify the file name. Use the field on the runtime tab to select a file for viewing. View the file by clicking **View**.

<b>Data type</b>	String
<b>Units</b>	File path name

## **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 > Application Servers >server\_name**. Then, under Server Infrastructure, click **Java and Process Management > Process Definition > Monitoring Policy**.

### ***Maximum Startup Attempts:***

Specifies the maximum number of times to attempt to start the application server before giving up.

<b>Data type</b>	Integer
------------------	---------

### ***Ping Interval:***

Specifies 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.

<b>Data type</b>	Integer
<b>Range</b>	Set the value greater than or equal to 0 (zero) and less than 2147483.

### ***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.

<b>Data type</b>	Integer
<b>Units</b>	Seconds
<b>Range</b>	Set the value greater than or equal to 0 (zero) and less than 2147483.

### ***Automatic Restart:***

Specifies whether the process should restart automatically if it fails. The default is to restart the process automatically.

<b>Data type</b>	Boolean
<b>Default</b>	true

### ***Node Restart State:***

Specifies the desired state for the process after the node completely shuts down and restarts.

<b>Data type</b>	String
<b>Default</b>	STOPPED
<b>Range</b>	Valid values are STOPPED, RUNNING, or PREVIOUS. If you want the process to return to its current state after the node restarts, use PREVIOUS.

## **Automatically restarting server processes**

There are several server processes related to WebSphere Application Server products that the operating system can monitor and automatically restart when the server processes stop abnormally. This task describes how to set up these *monitored* processes.

To set up this function on a Linux or UNIX-based operating system, you must have root authority to edit the inittab file.

On a Windows operating system, you must belong to the Administrator group and have the following advanced user rights:

- Act as part of the operating system

- Log on as a service

The Installation wizard grants you the user rights if your user ID is part of the administrator group. If you are running on a Microsoft Windows 2000 Operating System, the Installation wizard displays a message that states that although the advanced user rights are now effective, they do not display as effective until the next time you log on to the Windows machine.

You can also add the advanced user rights manually if you are performing a silent installation on a Windows platform. For example, to grant the user rights to your administrator group user ID on a Windows 2000 Server platform, perform the following procedure:

1. Click **Administrative Tools** in the Control Panel.
2. Click **Local Security Policy**.
3. Click **Local Policies**.
4. Click **User Rights Assignments**.
5. Right click **Act as part of the operating system**.
6. Click **Security**.
7. Click **Add**.
8. Click your user ID.
9. Click **Add**.
10. Click **OK**.
11. Click **OK**.
12. Right click **Log on as a service**.
13. Click **Security**.
14. Click **Add**.
15. Click **OK**.
16. Click **OK**.
17. Reboot your machine to make the settings effective.

Consult your Windows help system for more information.

There are several environments where you might use this function of automatically restarting servers. You can restart the **server1** managed node process, for example. Here is a list of processes you might consider restarting:

- The **server1** managed node process
- The **server1** process on a stand-alone Application Server
- The **dmgr** process on a deployment manager node
- The **nodeagent** server process on any managed node
- The **IBM HTTP Server** process
- The **IBM HTTP Administration** process

You can create Windows services during installation, using the installation wizard. The wizard lets you create services for these servers:

- The **server1** managed node process, defined as a manually started (versus automatic) service
- The **server1** stand-alone Application Server process, defined as a manually started service
- The **IBM HTTP Server** process and the **IBM HTTP Administration** process, defined as automatically started services when you choose to install the IBM HTTP Server feature
- The **dmgr** process on a deployment manager node, defined as a manually started service

The installation wizard does not provide a way to create a service for a node agent because the deployment manager instantiates each node agent after installation when you add an Application Server node to the deployment manager cell. For this reason, you must manually create a function that automatically starts a failed nodeagent server process.

You must manually create a shell script that automatically starts any of the processes previously mentioned, on a Linux and UNIX-based operating system. Each Windows service or UNIX shell script controls a single process, such as a stand-alone WebSphere Application Server instance. Multiple stand-alone Application Server processes require multiple Windows service or UNIX scripts, which you can define.

In a Network Deployment environment, the **addNode** or **startNode** command starts a single unmonitored node agent only, the nodeagent process, and does not start all of the processes that you might define on the node. While running, the node agent monitors and restarts Application Server processes on that node, on either a Windows or a Linux and UNIX-based platform. Each Application Server process has MonitoringPolicy configuration settings that the node agent uses when monitoring and restarting the process.

It is recommended that you set up a monitored node agent process manually, either through a Windows service, or through the rc.was example shell script on Linux and UNIX-based platforms. The operating system monitors and automatically restarts the node agent process, nodeagent, if the process terminates abnormally, which means if the process stops without going through a normal shutdown. Setting up the deployment manager server, dmgr, as a monitored process is recommended. As mentioned, you can do this during installation on a Windows platform. On a Linux and UNIX-based platform, use the rc.was example shell script to set up the deployment manager dmgr server as a monitored process.

If you do not install the WebSphere Application Server Network Deployment product as a Windows service during installation, you can use the do so at a later time. The operating system can then monitor each server process and restart the process if it stops.

1. **Use the profile creation wizard** to set up a Windows service to automatically monitor and restart processes related to the WebSphere Application Server product.

- Perform the following procedure from the profile creation wizard to select services that the installation wizard can set up:

a. Click **Run WebSphere Application Server Network Deployment as a service**.

If you select this option, the installation wizard creates the following service during installation:

**IBMWAS6Service** - *node\_name*

**IBMWAS6Service** -*node\_name* service controls the *node\_name* process.

After you complete and verify the installation, use the Windows Services panel to change the **IBMWAS6Service** -*node\_name* service to an automatic startup type.

1) Right click **IBMWAS6Service** -*node\_name* and click **Properties**.

2) Click **Automatic** from the **Startup type** list box and click **OK**.

b. Click **Run IBM HTTP Server as a service**.

Select this option on the machine where you are installing the IBM HTTP Server.

If you select this option, the installation wizard creates the following services during the installation:

– **IBM HTTP Server 2.0.x**

– **IBM HTTP Administration 2.0.x**

The installation wizard defines the startup type of these services as **automatic**. It is not necessary for you to change the type from manual to automatic.

c. Enter your user ID and password and click **Next**.

In a coexistence environment, you can change the default service names to make them unique. In a same version coexistence scenario for IBM HTTP Server 2.0.x on a Windows platform, you cannot use the default service names created by the installer because they are common.

To work around this problem:

a. Install the first copy of IBM HTTP Server, either by itself or with WebSphere Application Server and select to install the services.

b. Customize the service names for the first install by running the following commands from the first install location:

- ```

    apache -k install -n "IHS 2.0(1)"
    apache -k install -f conf\admin.conf -n "IHS 2.0 Administration (1)"

```
- c. Edit the AdminAlias directive in the *installLocation 1\conf\admin.conf* file to point to the new service name, such as **IHS 2.0(1)**.
  - d. Remove the default service names installed by the first install by running the following commands:
 

```

        apache -k uninstall -n "IBM HTTP Server 2.0"
        apache -k uninstall -n "IBM HTTP Administration 2.0"
      
```
  - e. Install the second copy of IBM HTTP Server, either by itself or with WebSphere Application Server. The default service names correspond to the second install.

**Note:** Customized service names must be unique on your system.

2. **After installing**, you can use the WASService.exe utility in the *install\_root\bin* directory to manually define a Windows service for another installation instance or for another configuration instance of the server1 process.
3. **After installing**, use the WASService.exe tool to manually define the nodeagent server process as a Windows service.  
You can use the same tool to manually define a Windows service for another installation instance or for another configuration instance of either the server1 process or the dmgr process.
4. **After installing**, set up a Linux and UNIX-based shell script to automatically monitor and restart the nodeagent process or any other related server process.
  - a. Locate the rc.was example shell script, which is in the *install\_root/bin* directory.
  - b. Create a new shell script for each process that the operating system is to monitor and restart.
  - c. Edit each shell script according to comments in its header, which provide instructions for identifying a WebSphere Application Server process.
  - d. Edit the inittab table of the operating system, to add an entry for each shell script you have created.

Comments in the header of the rc.was file show a sample inittab entry line for adding the script. This inittab entry causes the Linux and UNIX-based system to call each shell script whenever the system initializes. As it runs, each shell script monitors and starts the server process you specified.

Each shell script monitors and restarts the following processes in an WebSphere Application Server Network Deployment environment:

- A server process on a managed node
- A node agent process on a managed node
- A stand-alone Application Server process
- A deployment manager process

You can use the **net start** and **net stop** commands to control the IBM HTTP Server services on a Windows system. For more information about these commands, see the Windows help file. Access these commands from the Start menu, clicking **Start > Programs > IBM HTTP Server**.

You can also use the **Start the Server** and **Stop the Server** commands to control the IBM WebSphere Application Server process on a Windows system. Access these commands from the Start menu, clicking **Start > Programs > IBM WebSphere > Application Server V6**.

You can also use the **Start the Manager** and **Stop the Manager** commands to control the Network Deployment dmgr process on a Windows system. Access these commands from the Start menu, clicking **Start > Programs > IBM WebSphere > Application Server V6 > Deployment Manager**.

Processes started by a **startServer** command, a **startNode** command, or a **startManager** command are not running as monitored processes, regardless of how you have configured them.



For example, you can configure a server1 process as a monitored process. However, if you start the server1 process using the **startServer** command, the operating system does not monitor or restart the server1 process because the operating system did not originally start the process as a monitored process.

Return to Defining application server processes to continue.

## WASService command

The **WASService** command line tool lets you create a Windows service for any WebSphere Application Server Java process.

You can create Windows services for WebSphere Application Server Java processes. Potential Windows services include the following server processes:

- The default server1 process on an application server node
- Application server processes that you create on an application server node
- The nodeagent process on an application server node that is part of a deployment manager cell
- The deployment manager process, dmgr

**Location of the command file:** The WASService.exe command file is located in the *install\_root*\bin directory.

### Command syntax:

#### WASService.exe command syntax for starting an existing service

The command syntax is as follows:

```
WASService.exe [-start] "service_name" [optional startServer.bat parameters]
```

#### WASService.exe command syntax for creating a service or updating an existing service

The command syntax is as follows:

```
WASService.exe -add "service_name"  
                -serverName server  
                -profilePath server_profile_directory  
  
                [-wasHome install_root]  
                [-configRoot configuration_repository_directory]  
                [-startArgs additional_start_arguments]  
                [-stopArgs additional_stop_arguments]  
                [-userid user_id -password password]  
                [-logFile service_log_file]  
                [-logRoot server_log_directory]  
                [-restart true | -restart false]  
                [-startType automatic | manual | disabled]
```

#### WASService.exe command syntax for deleting a service

The command syntax is as follows:

```
WASService.exe -remove "service_name"
```

#### WASService.exe command syntax for stopping a running service

The command syntax is as follows:

```
WASService.exe -stop "service_name" [optional stopServer.bat parameters]
```

## WASService.exe command syntax for retrieving service status

The command syntax is as follows:

```
WASService.exe -status "service_name"
```

## WASService.exe command syntax for encoding parameters

The command syntax is as follows:

```
WASService.exe -encodeParams "service_name"
```

**Parameters:** Supported arguments include:

**-add "service\_name"**

Creates a service named *service\_name* or updates an existing Windows service. The syntax is the same for both cases.

**-configRoot configuration\_repository\_directory**

Optional parameter that identifies the configuration directory of the installation root directory of a WebSphere Application Server product.

**-encodeParams service\_name**

Optional parameter that forces the service to encode the `-startArgs` and `-stopArgs` so that the arguments cannot be determined by editing the registry. Use the parameter when creating a service with the `-add` parameter by adding `-encodeParams` to the command line with no arguments. Or encode the parameters of an existing service:

```
WASService -encodeParams service_name
```

**-logFile service\_log\_file**

Optional parameter that identifies a log file that the **WASService** command uses to record its activity.

**-logRoot server\_log\_directory**

Required parameter that identifies the server log directory for the profile. The **WASService** command looks for a file named *server\_name*.pid to determine if the server is running.

**-profilePath server\_profile\_directory**

Specifies the directory path of the profile that defines the server process.

**-remove service\_name**

Deletes the specified service.

**-restart true | false**

Restarts the existing service automatically if the service fails when set to true.

**-serverName Server\_name**

Identifies the server that the service controls.

**-start "service\_name" [optional startServer.bat parameters]**

Starts the existing service.

**-startArgs additional\_start\_arguments**

Optional parameter that identifies additional parameters.

**-startType automatic | manual | disabled**

Defines the startup type of the new service. An automatic startup type starts automatically when the system starts or when the service is called for the first time. You must start a manual service before the operating system can load it and make it available. You cannot start a disabled service before changing the startup type.

**-status service\_name**

Returns the current status of the service, which includes whether the service is running or stopped.

**-stop service\_name [optional stopServer.bat parameters]**

Stops the specified service.

**-stopArgs** *additional\_stop\_arguments*

Optional parameter that identifies additional parameters.

**-userid** *user\_ID* **-password** *password*

Optional parameters that identify a privileged user ID and password that the Windows service will run as.

**-wasHome** *install\_root*

Optional parameter that identifies the installation root directory of the WebSphere Application Server product.

**Default names for Windows services that are created by the wizard:** The names of the Windows services that the Profile creation wizard can create are:

### Deployment manager

IBM WebSphere Application Server V6 - *node\_name\_of\_the\_deployment\_manager\_node*

### Application Server

IBM WebSphere Application Server V6 - *node\_name\_of\_the\_server1\_node*

### Custom profile

After federating the node and creating an Application Server, a service can be created named IBM WebSphere Application Server V6 - *node\_name\_of\_the\_managed\_node*.

After creating a custom profile, you must federate the node to create a node agent server on the node. You can also use the administrative console of the deployment manager to create application server processes on the node. You can create a Windows service for the nodeagent server process or the application servers on the node.

A node agent server is also created after adding an application server node to a deployment manager cell. You can create a Windows service for the nodeagent server process as described later.

**Viewing the Windows services panel:** To view Windows services, open the Control panel and click **Administrative Tools > Services**. Select a service to view information about it. Right click the service and click **Properties**. Four tabs provide information and functionality. For example, select the **Setup type** field on the **General** tab to change the setup type.

### Examples:

#### Creating a deployment manager service

This example creates a service called *IBM WebSphere Application Server V6 - dmgr* that starts the dmgr process:

```
WASService -add dmgr
            -servername dmgr
            -profilePath "C:\Program Files\IBM\WebSphere\AppServer\
                        profiles\Dmgr1"
            -wasHome "C:\Program Files\IBM\WebSphere\AppServer"
            -logfile "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\Dmgr1\logs\startManager.log"
            -logRoot "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\Dmgr1\logs"
            -restart true
```

After entering the command, messages that are similar to those in the following example display in the command window:

```
Adding Service: dmgr
Config Root: C:\Program Files\IBM\WebSphere\AppServer\profiles\Dmgr1\config
Server Name: dmgr
Profile Path: C:\Program Files\IBM\WebSphere\AppServer\profiles\Dmgr1
Was Home: D:\Program Files\IBM\WebSphere\AppServer\
```

```
Start Args:
Restart: 1
IBM WebSphere Application Server V6 - dmgr service successfully added.
```

Click **Start > Settings > Control Panel > Administrative Tools > Services** to work with the new service.

### Creating a node agent service

This example creates a service called *IBM WebSphere Application Server V6 - nodeagent* that starts the nodeagent process:

```
WASService -add nodeagent
            -servername nodeagent
            -profilePath "C:\Program Files\IBM\WebSphere\AppServer\
                        profiles\CustomProfile"
            -wasHome "C:\Program Files\IBM\WebSphere\AppServer"
            -logfile "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\CustomProfile\logs\startNode.log"
            -logRoot "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\CustomProfile\logs"
            -restart true
```

After entering the command, messages that are similar to those in the following example display in the command window:

```
Adding Service: nodeagent
    Config Root: C:\Program Files\IBM\WebSphere\AppServer\
                profiles\CustomProfile\config
    Server Name: nodeagent
    Profile Path: C:\Program Files\IBM\WebSphere\AppServer\
                profiles\CustomProfile
    Was Home: C:\Program Files\IBM\WebSphere\AppServer\
    Start Args:
    Restart: 1
IBM WebSphere Application Server V6 - nodeagent service successfully added.
```

### Creating an Application Server service

This example creates a service called *IBM WebSphere Application Server V6 - server2* that starts an Application Server process:

```
WASService -add server2
            -servername server2
            -profilePath "C:\Program Files\IBM\WebSphere\AppServer\
                        profiles\CustomProfile"
            -wasHome "C:\Program Files\IBM\WebSphere\AppServer"
            -logfile "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\CustomProfile\logs\startNode.log"
            -logRoot "C:\Program Files\IBM\WebSphere\AppServer\
                    profiles\CustomProfile\logs"
            -restart true
```

After entering the command, messages that are similar to those in the following example display in the command window:

```
Adding Service: server2
    Config Root: C:\Program Files\IBM\WebSphere\AppServer\
                profiles\CustomProfile\config
    Server Name: server2
    Profile Path: C:\Program Files\IBM\WebSphere\AppServer\
                profiles\CustomProfile
    Was Home: C:\Program Files\IBM\WebSphere\AppServer\
    Start Args:
    Restart: 1
IBM WebSphere Application Server V6 - server2 service successfully added.
```

## Updating an existing Application Server service

This example updates an existing service called IBM WebSphere Application Server V6 - server2 with additional stop arguments, username and password. The user name and password are required by the **stopServer** command to stop the application server with security enabled.

```
WASService -add server2
            -servername server2
            -profilePath "C:\Program Files\IBM\WebSphere\AppServer\
                        profiles\CustomProfile"
            -stopArgs "-username user_name -password password"
            -encodeParams server2
```

**Starting and stopping a server process after creating a Windows service:** If you issue the **startServer server1** command or the **stopServer server1** after creating a Windows service for server1, a message that is similar to the following example displays:

Because server1 is registered to run as a Windows Service, the request to start this server will be completed by starting the associated Windows Service.

If you issue the **startNode** command or the **stopNode** command after creating a Windows service for the nodeagent process, a message that is similar to the following example displays:

Because nodeagent is registered to run as a Windows Service, the request to start or stop this server will be completed by starting or stopping the associated Windows Service. Examine the log files to view messages related to this command.

If you issue the **startManager** command or the **stopManager** command after creating a Windows service for the deployment manager, a message that is similar to the following example displays:

Because dmgr is registered to run as a Windows Service, the request to start or stop this server will be completed by starting or stopping the associated Windows Service. Examine the log files to view messages related to this command.

## Stopping a server after enabling security

If you enable security while a Windows service is running, you cannot stop the server from the command line, even when using the username and password parameters on the **stopServer** command. A message similar to the following example is displayed:

```
Could not stop the IBM WebSphere Application Server V6 -
server_name service on Local Computer. The service
did not return an error. This could be an internal Windows
error or an internal service error. If the problem persists,
contact your system administrator.
```

The problem is due to the service control of the process. You must change the service to use the proper stop-server arguments for a secure server.

Use the **-stopArgs** parameter and the **-encodeParams** parameter to update the service as described in the "Updating an existing application server service" example.

---

## Java virtual machines (JVMs)

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.

---

## Using the JVM

As part of configuring an application server, you might define settings that enhance your system's use of the Java virtual machine (JVM).

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 the wsadmin tool to change the configuration through scripting.

1. Access the Java Virtual Machine page.
  - a. Click **Servers > Application Servers** in the console navigation tree.
  - 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, click **Java and Process Management > Process Definition**.
  - d. On the Process Definition page, click **Java Virtual Machine**.
2. On the Java Virtual Machine page, specify values for the JVM settings as needed and click **OK**.
3. Click **Save** on the console task bar.
4. Restart the application server.

"“Configuring application servers for UTF-8 encoding” on page 193” 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 248” 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 for the application server's process.

To view this administrative console page, click **Servers > Application Servers > *server\_name* > Process Definition > Java Virtual Machine**.

### Classpath

Specifies the standard class path in which the Java virtual machine code looks for classes.

Enter each classpath entry into a table row. You do not need to add the colon or semicolon at the end of each entry.

|                  |            |
|------------------|------------|
| <b>Data type</b> | String     |
| <b>Units</b>     | Class path |

### Boot Classpath

Specifies bootstrap classes and resources for JVM code. This option is only available for JVM instructions that support bootstrap classes and resources. You can separate multiple paths by a colon (:) or semi-colon (;), depending on operating system of the node.

|                  |        |
|------------------|--------|
| <b>Data type</b> | String |
|------------------|--------|

## Verbose Class Loading

Specifies whether to use verbose debug output for class loading. The default is not to enable verbose class loading.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Boolean |
| <b>Default</b>   | false   |

## Verbose Garbage Collection

Specifies whether to use verbose debug output for garbage collection. The default is not to enable verbose garbage collection.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Boolean |
| <b>Default</b>   | false   |

## 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.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Boolean |
| <b>Default</b>   | false   |

## Initial Heap Size

Specifies the initial heap size available to the JVM code, in megabytes.

Increasing the minimum heap size can improve startup. The number of garbage collection occurrences are reduced and a 10% gain in performance is realized.

Increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory, in general. After the heap begins swapping to disk, Java performance suffers drastically.

|                  |                    |
|------------------|--------------------|
| <b>Data type</b> | Integer            |
| <b>Default</b>   | The default is 50. |

## Maximum Heap Size

Specifies the maximum heap size available to the JVM code, in megabytes.

Increasing the heap size can improve startup. By increasing heap size, you can reduce the number of garbage collection occurrences with a 10% gain in performance.

Increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory, in general. After the heap begins swapping to disk, Java performance suffers drastically. Set the maximum heap size low enough to contain the heap within physical memory.

|                  |                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Data type</b> | Integer                                                                                                               |
| <b>Default</b>   | 0 for iSeries, 256 for all other platforms. Keep the value low enough to avoid paging or swapping-out-memory-to-disk. |

## Debug Mode

Specifies whether to run the JVM in debug mode. The default is not to 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.



|                  |         |
|------------------|---------|
| <b>Data type</b> | Boolean |
| <b>Default</b>   | 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 Debug Mode is enabled.

Debug arguments are only required if the Debug Mode property is set to true. If you enable debugging on multiple application servers on the same node, make sure that the servers are using different address arguments, which define the port for debugging. For example, if you enable debugging on two servers and leave the default debug port for each server as `address=7777`, the servers could fail to start properly.

|                  |                             |
|------------------|-----------------------------|
| <b>Data type</b> | String                      |
| <b>Units</b>     | 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.

The following are optional command line arguments that you can use by entering them into the **General JVM Arguments** field. If you enter more than one argument, separate each argument by a space.

**Note:** If the argument says it is for the IBM Developer Kit only, you cannot use the argument with another JVM, such as the Sun JDK or the HP JDK.

- **-Xquickstart:** You can use **-Xquickstart** for initial compilation 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. 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%.

The **-Xquickstart** option is not supported on OS/400.

- **-Xverify:none:** When using this value, the class verification stage is skipped during class loading . By using **-Xverify:none** with the just in time (JIT) compiler enabled, startup time is improved by 10-15%.

The **-Xverify:none** option is not supported on OS/400.

- **-Xnoclassgc:** You can use this value to disable class garbage collection, which leads to more class reuse and slightly improved performance. The trade-off is that you won't be collecting the resources owned by these classes. You can monitor garbage collection using the `verbose:gc` configuration setting, which will output class garbage collection statistics. Examining these statistics will help you understand the trade-off between the reclaimed resources and the amount of garbage collection required to reclaim the resources. However, if the same set of classes are garbage collected repeatedly in your workload, you should disable garbage collection. Class garbage collection is enabled by default.
- **-Xgcthreads:** You can use several garbage collection threads at one time, also known as *parallel garbage collection*. When entering this value in the **Generic JVM Arguments** field, also enter the number of processors that your machine has, for example, `-Xgcthreads=number_of_processors`. On a node with *n* processors, the default number of threads is *n*. You should use parallel garbage collection if your machine has more than one processor. This argument is valid only for the IBM Developer Kit.

The **-Xgcthreads** option is not supported on OS/400.

- **-Xnocompactgc:** This value disables heap compaction which is the most expensive garbage collection operation. Avoid compaction in the IBM Developer Kit. If you disable heap compaction, you eliminate all associated overhead.
- **-Xinitsh:** You can use this value to set the initial heap size where class objects are stored. The method definitions and static fields are also stored with the class objects. Although the system heap size has no upper bound, set the initial size so that you do not incur the cost of expanding the system heap size, which involves calls to the operating system memory manager. You can compute a good initial system heap size by knowing the number of classes loaded in the WebSphere Application Server product,

which is about 8,000 classes, and their average size. Having knowledge of the applications helps you include them in the calculation. You can use this argument only with the IBM Developer Kit.

- **-Xgcpolicy:** You can use this value to set the garbage collection policy. If the garbage collection policy (gcpolicy) is set to optavgpause, concurrent marking is used to track application threads starting from the stack before the heap becomes full. The garbage collector pauses become uniform and long pauses are not apparent. The trade-off is reduced throughput because threads might have to do extra work. The default, recommended value is optthruput. Enter the value as `-Xgcpolicy:[optthruput|optavgpause]`. You can use this argument only with the IBM Developer Kit.
- **-XX:** The Sun-based Java Development Kit (JDK) Version 1.4.2 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 short lived objects so that objects in the pool do not live through more than one garbage collection cycle. The size of new generation pool is determined by the `NewSize` and `MaxNewSize` parameters. Objects that survive the first garbage collection cycle are transferred to another pool. The size of the survivor pool is determined by parameter `SurvivorRatio`. If garbage collection becomes a bottleneck, you can try customizing the generation pool settings. To monitor garbage collection statistics, use the object statistics in Tivoli Performance Viewer or the verbose:gc configuration setting. Enter the following values: `-XX:NewSize` (lower bound) , `-XX:MaxNewSize` (upper bound), and `-XX:SurvivorRatio=NewRatioSize`. The default values are:`NewSize=2m MaxNewSize=32m SurvivorRatio=2` However, if you have a JVM with more than 1 GB heap size, you should use the values: `-XX:newSize=640m -XX:MaxNewSize=640m -XX:SurvivorRatio=16`, or set 50 to 60% of total heap size to a new generation pool.

The **-XX** option is not supported on OS/400.

- **-Xminf:** You can use this value to specify the minimum free heap size percentage. The heap grows if the free space is below the specified amount. In reset enabled mode, this option specifies the minimum percentage of free space for the middleware and transient heaps. This is a floating point number, 0 through 1. The default is .3 (30%).
- **-server | -client:** Java HotSpot Technology in the Sun-based Java Development Kit (JDK) Version 1.4.2 introduces an adaptive JVM containing algorithms for optimizing byte code execution over time. The JVM runs in two modes, **-server** and **-client**. If you use the default **-client** mode, there will be a faster startup time and a smaller memory footprint, but lower extended performance. You can enhance performance by using **-server** mode if a sufficient amount of time is allowed for the HotSpot JVM to warm up by performing continuous execution of byte code. In most cases, use **-server** mode, which produces more efficient run-time execution over extended periods. You can monitor the process size and the server startup time to check the difference between **-client** and **-server**.

The **-server | -client** option is not supported on OS/400.

|                  |                             |
|------------------|-----------------------------|
| <b>Data type</b> | String                      |
| <b>Units</b>     | Java command line arguments |

### Executable JAR File Name

Specifies a full path name for an executable JAR file that the JVM code uses.

|                  |           |
|------------------|-----------|
| <b>Data type</b> | String    |
| <b>Units</b>     | 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.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Boolean |
|------------------|---------|

|                    |                     |
|--------------------|---------------------|
| <b>Default</b>     | false (JIT enabled) |
| <b>Recommended</b> | JIT enabled         |

## Operating System Name

Specifies JVM settings for a given operating system. When started, the process uses the JVM settings for the operating system of the node.

|                  |        |
|------------------|--------|
| <b>Data type</b> | String |
|------------------|--------|

## 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*. To instruct the server not to use the Web server's document root and to use instead the Web application's context root for `sendRedirect()` calls, configure the 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. Click **Servers > Application Servers** in the console navigation tree.
  - 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, under Server Infrastructure, click **Java and Process Management > Process Definition**.
  - d. On the Process Definition page, click **Java Virtual Machine**.
  - e. On the Java Virtual Machine page, click **Custom Properties**.
  - f. On the Custom Properties page, click **New**.
2. On the settings page for a property, specify a name of `com.ibm.websphere.sendredirect.compatibility` and either `true` or `false` for the value, then click **OK**.
3. Click **Save** on the console task bar.
4. Stop the application server and then restart the application server.

## Setting custom JVM properties

In the WebSphere Application Server administrative console, you can change the values of the following custom JVM properties:

### **com.ibm.websphere.network.useMultiHome**

#### **For a distributed platform:**

Set this property in a multihomed environment where WebSphere Application Server is restricted to listen only on a specific IP address for Discovery and SOAP messages.

The settings for the `com.ibm.websphere.network.useMultiHome` property are as follows:

- Setting this property to `false` specifies that WebSphere Application Server will listen on all IP addresses on the host for Discovery and SOAP messages.
- Setting this property to `true` specifies that WebSphere Application Server will only listen on the configured host name for Discovery and SOAP messages. If you set this property to `true`, you should have a host name configured on WebSphere Application Server that resolves to a specific IP address.
- Setting this property to `null` specifies that WebSphere Application Server will only listen on the default IP address only.

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. In order for these changes to take place, you must restart the server.

**Steps for this task**

1. To set this property, connect to the administrative console and navigate to the indicated page.

|                    |                                                                                                                                                          |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Application server | <b>Servers &gt; Application Servers &gt; <i>server1</i> &gt; Process Definition &gt; Control &gt; Java Virtual Machine &gt; Custom Properties</b>        |
| Deployment manager | <b>System Administration &gt; Deployment Manager &gt; Process definition &gt; Control &gt; Java Virtual Machine &gt; Custom Properties</b>               |
| Node agent         | <b>System Administration &gt; Node Agent &gt; <i>nodeagent</i> &gt; Process definition &gt; Control &gt; Java Virtual Machine &gt; Custom Properties</b> |

2. If the **com.ibm.websphere.network.useMultiHome** property is not present in the list, create a new property name and indicate its value.
3. Restart the server.

**com.ibm.websphere.deletejspclasses**

Deletes JavaServer Pages classes for all applications after those applications have been deleted or updated. By default, the value of this property is true.

**Steps for this task**

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

**For a distributed platform:**

|                    |                                                                                                                                                                                                                           |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Base configuration | <b>Servers &gt; Application Servers &gt; <i>server1</i>. Then, under Server Infrastructure, click <b>Java and Process Management &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b></b>         |
| ND configuration   | <b>System Administration &gt; Node Agents &gt; <i>nodeagent</i>. Then, under Server Infrastructure, click <b>Java and Process Management &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b></b> |

2. If the **com.ibm.websphere.deletejspclasses** property is not present in the list, create a new property name.
3. Enter the name and value.

**com.ibm.websphere.deletejspclasses.delete**

Deletes JavaServer Pages classes for all applications after those applications have been deleted, but not after they have been updated. By default, the value of this property is true.

**Steps for this task**

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

**For a distributed platform:**

|                    |                                                                                                                                      |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Base configuration | <b>Servers &gt; Application Servers &gt; <i>server1</i> &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b> |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------|

|                  |                                                                                                                                                                                                                              |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ND configuration | <b>System Administration &gt; Node Agents &gt; <i>nodeagent</i>.</b><br>Then, under Server Infrastructure, click <b>Java and Process Management &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b> |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

2. If the **com.ibm.websphere.deletejspclasses.delete** property is not present in the list, create a new property name.
3. Enter the name and value.

### **com.ibm.websphere.deletejspclasses.update**

Deletes JavaServer Pages classes for all applications after those applications have been updated, but not after they have been deleted. By default, the value of this property is true.

#### **Steps for this task**

1. Connect to the administrative console and navigate to the Java Virtual Machine Custom Properties panel.

#### **For a distributed platform:**

|                    |                                                                                                                                                                                                                              |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Base configuration | <b>Servers &gt; Application Servers &gt; <i>server1</i>.</b> Then, under Server Infrastructure, click <b>Java and Process Management &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b>            |
| ND configuration   | <b>System Administration &gt; Node Agents &gt; <i>nodeagent</i>.</b><br>Then, under Server Infrastructure, click <b>Java and Process Management &gt; Process definition &gt; Java Virtual Machine &gt; Custom Properties</b> |

2. If the **com.ibm.websphere.deletejspclasses.update** property is not present in the list, create a new property name.
3. Enter the name and value.

## **Tuning Java virtual machines**

The application server, being a Java process, requires a Java virtual machine (JVM) to run, and to support the Java applications running on it. As part of configuring an application server, you can fine-tune settings that enhance system use of the JVM. In addition to the following tuning parameters, see also “Java memory tuning tips” on page 251.

Use the following JVM parameters, including garbage collection options for IBM Developer Kit 1.4.2, to tune the Java virtual machine. For instructions on view and change the JVM configuration, go to “Using the JVM” on page 244. For information on specifying any of the following settings, go to “Java virtual machine settings” on page 244.

- **Specify any or all of the following generic JVM arguments.** These optional command line arguments are passed to the Java virtual machine code that starts the application server process.
  - Quickstart (-Xquickstart)
  - Avoiding class verification (-Xverify:none)
  - Class garbage collection (-Xnoclassgc)
  - Garbage collection threads (-Xgcthreads)
  - Garbage collection policy (-Xgcpolicy)
  - Sun JDK 1.4.2 Generational Garbage Collection (-XX)

You can find more information about generational garbage collection at <http://java.sun.com/docs/hotspot/gc/index.html>.

  - Sun Java Development Kit 1.4.2 HotSpot JVM warm-up (-server)
  - Heap compaction (-Xnocompactgc)
  - Initial system heap size (-Xinitsh)

- **Set the initial heap size.**
- **Set the maximum heap size.**

---

## 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.

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 run-time environment to support applications.

1. Create an application server.
2. Create a virtual host.
3. Configure a Web container. See the *Developing and deploying applications* PDF for information on how to perform this task.
4. Configure an EJB container. See the *Developing and deploying applications* PDF for information on how to perform this task.
5. Create resources for data access. See the *Developing and deploying applications* PDF for information on how to perform this task.
6. Create a JDBC provider and data source. See the *Developing and deploying applications* PDF for information on how to perform this task.
7. Create a URL and URL provider. See the *Developing and deploying applications* PDF for information on how to perform this task.
8. Create a JavaMail session. See the *Developing and deploying applications* PDF for information on how to perform this task.
9. Create resources for session support. See the *Developing and deploying applications* PDF for information on how to perform this task.
10. Configure a Session Manager. See the *Developing and deploying applications* PDF for information on how to perform this task.

---

## Java memory tuning tips

Enterprise applications written in the Java language involve complex object relationships and utilize large numbers of objects. Although, the Java language automatically manages memory associated with object life cycles, understanding the application usage patterns for objects is important. In particular, verify the following:

- The application is not over-utilizing objects
- The application is not leaking objects
- The Java heap parameters are set properly to handle a given object usage pattern

Understanding the effect of garbage collection is necessary to apply these management techniques.

### The garbage collection bottleneck

Examining Java garbage collection gives insight to how the application is utilizing memory. Garbage collection is a Java strength. By taking the burden of memory management away from the application writer, Java applications are more robust than applications written in languages that do not provide garbage collection. This robustness applies as long as the application is not abusing objects. Garbage collection normally consumes from 5% to 20% of total execution time of a properly functioning application. If not managed, garbage collection is one of the biggest bottlenecks for an application, especially when running on symmetric multiprocessing (SMP) server machines. The Java virtual machine (JVM) uses a parallel garbage collector to fully exploit an SMP during most garbage collection cycles where the Sun



HotSpot 1.3.1 JVM has a single-threaded garbage collector. For more information about garbage collection in a Solaris operating environment see the *Troubleshooting and support* PDF.

### The garbage collection gauge

You can use garbage collection to evaluate application performance health. By monitoring garbage collection during the execution of a fixed workload, you gain insight as to whether the application is over-utilizing objects. Garbage collection can even detect the presence of memory leaks.

You can monitor garbage collection statistics using object statistics in the Tivoli Performance Viewer, or using the **verbose:gc** JVM configuration setting. The **verbose:gc** format is not standardized between different JVMs or release levels. For a description of the IBM verbose:gc output and more information about the IBM garbage collector, see the *Troubleshooting and support* PDF.

For this type of investigation, set the minimum and maximum heap sizes to the same value. Choose a representative, repetitive workload that matches production usage as closely as possible, user errors included.

To ensure meaningful statistics, run the fixed workload until the application state is steady. It usually takes several minutes to reach a steady state.

### Detecting over-utilization of objects

You can use the Tivoli Performance Viewer to check if the application is overusing objects, by observing the counters for the JVM runtime. You have to set the **-XrunpmiJvmpiProfiler** command line option, as well as the JVM module maximum level in order to enable the Java virtual machine profiler interface (JVMPi) counters. The best result for the average time between garbage collections is at least 5-6 times the average duration of a single garbage collection. If you do not achieve this number, the application is spending more than 15% of its time in garbage collection.

If the information indicates a garbage collection bottleneck, there are two ways to clear the bottleneck. The most cost-effective way to optimize the application is to implement object caches and pools. Use a Java profiler to determine which objects to target. If you can not optimize the application, adding memory, processors and clones might help. Additional memory allows each clone to maintain a reasonable heap size. Additional processors allow the clones to run in parallel.

### Detecting memory leaks

Memory leaks in the Java language are a dangerous contributor to garbage collection bottlenecks. Memory leaks are more damaging than memory overuse, because a memory leak ultimately leads to system instability. Over time, garbage collection occurs more frequently until the heap is exhausted and the Java code fails with a fatal Out of Memory exception. Memory leaks occur when an unused object has references that are never freed. Memory leaks most commonly occur in collection classes, such as Hashtable because the table always has a reference to the object, even after real references are deleted.

High workload often causes applications to crash immediately after deployment in the production environment. This is especially true for leaking applications where the high workload accelerates the magnification of the leakage and a memory allocation failure occurs.

The goal of memory leak testing is to magnify numbers. Memory leaks are measured in terms of the amount of bytes or kilobytes that cannot be garbage collected. The delicate task is to differentiate these amounts between expected sizes of useful and unusable memory. This task is achieved more easily if the numbers are magnified, resulting in larger gaps and easier identification of inconsistencies. The following list contains important conclusions about memory leaks:

- **Long-running test**



Memory leak problems can manifest only after a period of time, therefore, memory leaks are found easily during long-running tests. Short running tests can lead to false alarms. It is sometimes difficult to know when a memory leak is occurring in the Java language, especially when memory usage has seemingly increased either abruptly or monotonically in a given period of time. The reason it is hard to detect a memory leak is that these kinds of increases can be valid or might be the intention of the developer. You can learn how to differentiate the delayed use of objects from completely unused objects by running applications for a longer period of time. Long-running application testing gives you higher confidence for whether the delayed use of objects is actually occurring.

- **Repetitive test**

In many cases, memory leak problems occur by successive repetitions of the same test case. The goal of memory leak testing is to establish a big gap between unusable memory and used memory in terms of their relative sizes. By repeating the same scenario over and over again, the gap is multiplied in a very progressive way. This testing helps if the number of leaks caused by the execution of a test case is so minimal that it is hardly noticeable in one run.

You can use repetitive tests at the system level or module level. The advantage with modular testing is better control. When a module is designed to keep the private module without creating external side effects such as memory usage, testing for memory leaks is easier. First, the memory usage before running the module is recorded. Then, a fixed set of test cases are run repeatedly. At the end of the test run, the current memory usage is recorded and checked for significant changes. Remember, garbage collection must be suggested when recording the actual memory usage by inserting `System.gc()` in the module where you want garbage collection to occur, or using a profiling tool, to force the event to occur.

- **Concurrency test**

Some memory leak problems can occur only when there are several threads running in the application. Unfortunately, synchronization points are very susceptible to memory leaks because of the added complication in the program logic. Careless programming can lead to kept or unreleased references. The incident of memory leaks is often facilitated or accelerated by increased concurrency in the system. The most common way to increase concurrency is to increase the number of clients in the test driver.

Consider the following points when choosing which test cases to use for memory leak testing:

- A good test case exercises areas of the application where objects are created. Most of the time, knowledge of the application is required. A description of the scenario can suggest creation of data spaces, such as adding a new record, creating an HTTP session, performing a transaction and searching a record.
- Look at areas where collections of objects are used. Typically, memory leaks are composed of objects within the same class. Also, collection classes such as `Vector` and `Hashtable` are common places where references to objects are implicitly stored by calling corresponding insertion methods. For example, the `get` method of a `Hashtable` object does not remove its reference to the retrieved object.

Tivoli Performance Viewer can help find memory leaks. For best results, repeat experiments with increasing duration, like 1000, 2000, and 4000-page requests. The Tivoli Performance Viewer graph of used memory should have a sawtooth shape. Each drop on the graph corresponds to a garbage collection. There is a memory leak if one of the following occurs:

- The amount of memory used immediately after each garbage collection increases significantly. The sawtooth pattern looks more like a staircase.
- The sawtooth pattern has an irregular shape.

Also, look at the difference between the number of objects allocated and the number of objects freed. If the gap between the two increases over time, there is a memory leak.

Heap consumption indicating a possible leak during a heavy workload (the application server is consistently near 100% CPU utilization), yet appearing to recover during a subsequent lighter or near-idle workload, is an indication of heap fragmentation. Heap fragmentation can occur when the JVM can free sufficient objects to satisfy memory allocation requests during garbage collection cycles, but the JVM does not have the time to compact small free memory areas in the heap to larger contiguous spaces.

Another form of heap fragmentation occurs when small objects (less than 512 bytes) are freed. The objects are freed, but the storage is not recovered, resulting in memory fragmentation until a heap compaction has been run.

To avoid heap fragmentation, turn on the `-Xcompactgc` flag in the JVM advanced settings command line arguments. The `-Xcompactgc` function verifies that each garbage collection cycle eliminates fragmentation. However, compaction is a relatively expensive operation. See the heap compaction command line argument (`-Xnocompactgc`) in “Java virtual machine settings” on page 244 for more information.

## Java heap parameters

The Java heap parameters also influence the behavior of garbage collection. Increasing the heap size supports more object creation. Because a large heap takes longer to fill, the application runs longer before a garbage collection occurs. However, a larger heap also takes longer to compact and causes garbage collection to take longer. Refer to the sections on Java Heap sizes in “Java virtual machine settings” on page 244 for more information.

*For performance analysis, the initial and maximum heap sizes should be equal.*

When tuning a production system where the working set size of the Java application is not understood, a good starting value for the initial heap size is 25% of the maximum heap size. The JVM then tries to adapt the size of the heap to the working set size of the application.

The illustration represents three CPU profiles, each running a fixed workload with varying Java heap settings. In the middle profile, the initial and maximum heap sizes are set to 128MB. Four garbage collections occur. The total time in garbage collection is about 15% of the total run. When the heap parameters are doubled to 256MB, as in the top profile, the length of the work time increases between garbage collections. Only three garbage collections occur, but the length of each garbage collection is also increased. In the third profile, the heap size is reduced to 64MB and exhibits the opposite effect. With a smaller heap size, both the time between garbage collections and the time for each garbage collection are shorter. For all three configurations, the total time in garbage collection is approximately 15%. This example illustrates an important concept about the Java heap and its relationship to object utilization. There is always a cost for garbage collection in Java applications.

Run a series of test experiments that vary the Java heap settings. For example, run experiments with 128MB, 192MB, 256MB, and 320MB. During each experiment, monitor the total memory usage. If you expand the heap too aggressively, paging can occur. Use the `vmstat` command or the Windows NT or Windows 2000 Performance Monitor to check for paging. If paging occurs, reduce the size of the heap or add more memory to the system. When all the runs are finished, compare the following statistics:

- Number of garbage collection calls
- Average duration of a single garbage collection call
- Ratio between the length of a single garbage collection call and the average time between calls

If the application is not over-utilizing objects and has no memory leaks, the state of steady memory utilization is reached. Garbage collection also occurs less frequently and for short duration.

If the heap free space settles at 85% or more, consider decreasing the maximum heap size values because the application server and the application are under-utilizing the memory allocated for heap.

For more information about garbage collection see the *Troubleshooting and support* PDF.

For current information available from IBM Support on known problems and their resolution, see the IBM Support page.

IBM Support has documents that can save you time gathering information needed to resolve this problem. Before opening a PMR, see the IBM Support page.

---

## Configuring multiple network interface card support

Use this task to first prepare your server for multiple network interface card support and enable multiple network interface card logic, including configuring multiple network interface cards to coexist on the same machine.

1. Configure object request broker (ORB) properties to support multiple network interface cards.
  - Set the `com.ibm.CORBA.LocalHost` property to resolve to a valid host name or IP address. The value should not resolve to a local host or loop back address (for example, 127.0.0.1).
  - a. In the administrative console, click **Servers > Application servers > *server\_name* > Java and process management > Process Definition > Java Virtual Machine**.
  - b. Add the following generic JVM argument: Add this argument as a single line. It is split here for printing purposes.

```
-Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.ibm.ws.orb.transport.useMultiHome=false
```
2. Update the host name for all HTTP transport chains. In the administrative console, click **Servers > Application servers > *server\_name* > Web container settings > Web container transport chains**. Update the host name for all of the transport chains listed on this page. The value should not resolve to a local host or loop back address (for example, 127.0.0.1).
3. Change the initial state to **stopped** for each of the listener ports. In the administrative console, click:
  - **Servers > Application servers > *server\_name* > Messaging > Message listener service > Listener ports > SamplePtoPListenerPort**
  - **Servers > Application servers > *server\_name* > Messaging > Message Listener Service > Listener Ports > SamplePubSubListenerPort**Update the state to **stopped** for each port.
4. In a Network Deployment environment, you must also add the following generic Java virtual machine (JVM) arguments and make sure all nodes are in sync.
  - a. Add the following generic JVM argument. In the administrative console, click **System administration > Deployment manager > Java and process management > Process definition > Java virtual machine**, and add the following argument as a single line. It is split here for printing purposes.

```
-Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.ibm.ws.orb.transport.useMultiHome=false
```
  - b. Add the following generic JVM argument. In the administrative console, click **System administration > Node agents > *node\_agent\_server* > Java and process management > Process definition > Java virtual machine**, and add the following argument as a single line. It is split here for printing purposes.

```
-Dcom.ibm.CORBA.LocalHost=xxxx.xx.xx.xx -Dcom.ibm.websphere.network.useMultiHome=false -Dcom.ibm.CORBA.ServerSocketQueueDepth=50 -Dcom.ibm.ws.orb.transport.useMultiHome=false
```
5. Change the initial state of the JMS server to stopped. In the administrative console, click **Servers > JMS Servers > *JMS\_server* > Initial state > stopped**.
6. Apply these changes to all application servers and the deployment manager.

By completing these steps, you enabled multiple network interface card support.

To configure multiple application servers to coexist on a single machine that is using two network interface cards, perform the following steps:

1. Install the WebSphere Application Server base product for each network interface card. To install on distributed platforms, see the *Installing your application serving environment* PDF for more information.
2. Start the server that is on the first network interface card. Follow the preceding steps in this task to prepare this server for multiple network interface card support and enable multiple network interface card logic. After you complete this step, stop the server.
3. Start the server that is on the other network interface card. Follow the preceding steps in this task to prepare this server for multiple network interface card support and enable multiple network interface card logic. After you complete this step, stop the server.
4. Start the servers on both network interface cards.

By completing these steps, you enabled multiple application servers to coexist on a single machine that has two network interface cards.

---

## Tuning application servers

The WebSphere Application Server contains interrelated components that must be harmoniously tuned to support the custom needs of your end-to-end e-business application.

This group of interrelated components is known as the queuing network. The queuing network helps the system achieve maximum throughput while maintaining the overall stability of the system.

The follow steps describe various tuning tasks that may improve your application server performance. You can choose to implement any of these application server settings.

- **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 tune the ORB with the following parameters:
  - Set **Pass by reference (com.ibm.CORBA.iiop.noLocalCopies)** as described in the *Developing and deploying applications* PDF.
  - Set the **Connection cache minimum (com.ibm.CORBA.MaxOpenConnections)** as described in the *Developing and deploying applications* PDF.
  - Set **Maximum size** as described in the *Developing and deploying applications* PDF.
  - Set **com.ibm.CORBA.ServerSocketQueueDepth** as described in the *Developing and deploying applications* PDF.
  - Set the **com.ibm.CORBA.FragmentSize** as described in the *Developing and deploying applications* PDF.

The the *Developing and deploying applications* PDF offer tips on using these parameters to tune the ORB.

- **Tune the XML parser definitions.**
  - **Description:** Facilitates server startup by adding XML parser definitions to the `jaxp.properties` and `xerxes.properties` files in the `${install_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
- **Tune the dynamic cache service.** Using the dynamic cache service can improve performance. See the *Developing and deploying applications* PDF for information about using the dynamic cache service and how it can affect your application server performance.

- **Tune the Web container.** The WebSphere Application Server Web container manages all HTTP requests to servlets, JSPs and Web services. Requests flow through a transport chain to the Web container. The transport chain defines the important tuning parameters for performance for the Web container. There is a transport chain for each TCP port that WebSphere Application Server is listening on for HTTP requests. For example, the default HTTP port 9080 is defined in Web container inbound channel chain. Use the following parameters to tune the Web container:
  - HTTP requests are processed by a pool of server threads. The minimum and maximum thread pool size for the Web container can be configured for optimal performance. Generally, 5 to 10 threads per server CPU will provide the best throughput. The number of threads configured does not represent the number of requests WebSphere can process concurrently. Requests are queued in the transport chain when all threads are busy. To specify the thread pool settings:
    1. Click **Servers > Application Servers > *server\_name* Web Container Settings > Web Container > Web container transport chains.**
    2. Select the normal inbound chain for serving requests. This will usually be named WCInboundDefault, on port 9080.
    3. Click **TCP Inbound Channel (TCP\_2).**
    4. Set **Thread Pools** under Related Items.
    5. Select **WebContainer.**
    6. Enter values for **Minimum Size** and **Maximum Size.**
  - The HTTP 1.1 protocol provides a "keep-alive" feature to enable the TCP connection between HTTP clients and the server to remain open between requests. By default WebSphere Application Server will close a given client connection after a number of requests or a timeout period. After a connection is closed, it will be recreated if the client issues another request. Early closure of connections can reduce performance. Enter a value for the maximum number of persistent requests to (keep-alive) to specify the number of requests that are allowed on a single HTTP connection. Enter a value for persistent timeouts to specify the amount of time, in seconds, that the HTTP transport channel allows a socket to remain idle between requests. To specify values for Maximum persistent requests and Persistent timeout:
    1. Click **Servers > Application Servers > *server\_name* Web Container Settings > Web Container > Web container transport chains.**
    2. Select the normal inbound chain for serving requests. This will usually be named WCInboundDefault, on port 9080.
    3. Click **HTTP Inbound Channel (HTTP\_2).**
    4. Enter values for **Maximum persistent requests** and **Persistent timeout.**
- **Tune the EJB container.** An 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 *Developing and deploying applications* PDF.
  - **Break CMP enterprise beans into several enterprise bean modules** while Assembling EJB modules. See the *Developing and deploying applications* PDF for more information on how to perform this task.
- **Tune the session management.** The installed default settings for session management are optimal for performance. See the *Developing and deploying applications* PDF for more information about tuning session management.
- **Tune the data sources.** A data source is used to access data from the database. The following parameters reveal how the number of physical connections within a connection pool can change performance.
  - Set the **Maximum connection pool** and **Minimum connection pool** as described in the *Developing and deploying applications* PDF.
  - Set the **Statement cache size** as described in the *Developing and deploying applications* PDF.



---

## 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 allowing 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 *Developing and deploying applications* 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 `inProcessLogFilenamePrefix` 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 `inProcessLogFilenamePrefix` 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`.

---

## Application servers: Resources for learning

Use the following links to find relevant supplemental information about configuring application servers. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information about:

- Programming instructions and examples
- Programming specifications
- Administration

**Programming instructions and examples**

- WebSphere Application Server education at <http://www.ibm.com/software/webservers/learn/>.

**Programming specifications**

- The Java™ Virtual Machine Specification, Second Edition at <http://java.sun.com/docs/books/vmspec/>.
- Sun's technology forum for the Java™ Virtual Machine Specification at <http://forum.java.sun.com/forum.jsp?forum=37>

**Administration**

- Listing of all IBM WebSphere Application Server Redbooks at <http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere>.





---

## Chapter 12. Balancing workloads with clusters

Consider your options for configuring application servers. See “Managing application servers” on page 194 for more information.

To monitor application servers and manage the workloads of servers, use server clusters and cluster members.

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 to remote machines.

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. See “Replicating data across application servers in a cluster” on page 276 for more information. You can create a replication domain when creating a cluster.
3. Deploy the application onto the application server. See the *Developing and deploying applications* PDF for information on how to perform this task.
4. 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. See “Creating clusters” on page 264 for more information.
5. You can create one or more cluster members of the cluster.
6. Configure a backup cluster that handles requests if the primary cluster fails.
7. Start all of the application servers by starting the cluster. Workload management automatically begins when you start the cluster members of the application server.
8. Once you have the cluster running, you can perform the following tasks:
  - Stop the cluster.
  - Upgrade applications on clusters. (See the *Developing and deploying applications* PDF for information on how to perform this task.)
  - Detect and handle problems with server clusters and their workloads.
  - Tune the behavior of the workload management run time. If your application is experiencing problems with timeouts or your network experiences extreme latency, change the timeout interval for the `com.ibm.CORBA.RequestTimeout` property. Or, if the workload management state of the client is refreshing too soon or too late, change the interval for the `com.ibm.websphere.wlm.unusable.interval` property.

For stand-alone Java clients, you must define a bootstrap host. Stand-alone Java clients are clients that are located on a different machine from the application server and have no administrative server. Add the following line to the Java Virtual Machine (JVM) arguments for the client:

```
-Dcom.ibm.CORBA.BootstrapHost=machine_name
```

where *machine\_name* is the name of the machine on which the administrative server is running.

---

### Clusters and workload management

Clusters are sets of servers that are managed together and participate in workload management. The servers that are members of a cluster can be on different host machines, as opposed to the servers that are part of the same node and must be located on the same host machine. A cell can have 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.

See Chapter 12, “Balancing workloads with clusters,” on page 261 to specify the amount of work targeted to each cluster member. You can distribute client tasks according to the capacities of different machines in the enterprise. A Web server plug-in routes application access among cluster members by server weighting, to provide better distribution control.

WebSphere Application Server can respond to increased use of an enterprise application by automatically replicating the application to additional cluster members as needed. This lets you deploy an application on a cluster instead of on a single node, without considering workload.

Multiple servers that can service the same client request is 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. In fact, several servers could fail, and as long as at least one cluster member is running, client requests can continue to be serviced. For further information backing up failed processes, see “Replicating data across application servers in a cluster” on page 276.

See “Backup clusters” on page 272 for information on backup cluster support. A backup cluster continues functioning when all cluster members of the primary cluster are not available.

---

## Clusters and node groups

Node groups bound clusters. All cluster members of a given cluster must be members of the same node group.

Any application you install to a cluster must be able to execute on any application server that is a member of that cluster. For the application you deploy to run successfully, all the members of the cluster must be located on nodes that meet the requirements for that application.

In a cell that has many different server configurations, it might be difficult to determine which nodes have the capabilities to host your application. A *node group* can be used to define groups of nodes that have enough in common to host members of a given cluster. All cluster members in a cluster must be in the same node group.

All nodes are members of at least one node group. When you create a cluster, the first application server you add to the cluster defines the node group that bounds the cluster. All other cluster members you add to the cluster can only be on nodes that are members of this same node group. When you create a new

cluster member in the administrative console, you are allowed to create the application server on a node that is a member of the node group for that cluster only.

Nodes can be members of multiple node groups. If the first cluster member you add to a cluster has multiple node groups defined, the system automatically chooses the node group that bounds the cluster. You can change the node group by modifying the cluster settings. Use the “Server cluster settings” on page 268 page to change the node group.

---

## Workload management (WLM) for distributed platforms

Workload management optimizes the distribution of client processing tasks. Incoming work requests are distributed 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.

Workload management provides the following benefits to WebSphere Application Server applications:

- It balances client workloads, allowing processing tasks to be distributed according to the capacities of the different machines in the system.
- 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 WebSphere Application Server environment, you implement workload management by using clusters, transports, and replication domains.

## 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 reply 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 WebSphere Application Server 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.
- The session manager, which is part of each application server, stores client session information and takes session affinity and server affinity into account when directing client requests to the cluster members of an application server. The workload management service considers server affinity and transaction affinity when directing client requests among the cluster members of an application server.

---

## Creating clusters

Use this task to create clusters, which are sets of application servers that are managed together and participate in workload management.

You can manage application servers collectively using a cluster. Use the “Server cluster collection” on page 267 to view and manage the cluster.

### 1. Enter basic cluster information.

- a. In the administrative console, click **Servers > Clusters > New**.
- b. Create a name for the cluster.
- c. To enable or disable node-scoped routing optimization, select **Prefer local**. The default is enabled, which indicates that, if possible, Enterprise JavaBeans (EJB) requests are routed to the client node. If you enable this feature, performance is improved because client requests are sent to local enterprise beans.
- d. To create a replication domain for this cluster, select **Create a replication domain for this cluster**. Use replication domains to transfer data, objects, or events for session manager, dynamic cache, or stateful session beans among the application servers in a cluster. Create a separate replication domain to use with each component that acts as a consumer of the replication. For example, you can configure one replication domain to use with a session manager and another domain to use with dynamic cache. The replication domain name that is created is identical to the cluster name. See “Replicating data across application servers in a cluster” on page 276 for more information.
- e. Choose whether to create an empty cluster or to create a cluster based on an existing server.  
To create an empty cluster, select **Do not include an existing server in this cluster**.  
To create a cluster based on an existing server, choose **Select an existing server to add to this cluster** and select the server you want to add. The server you add becomes a template for any additional cluster members that you add to the cluster. Be sure that the template is configured correctly before adding more cluster members that are based on this template. Note that this is the only time you are able to add an existing server to the cluster. After you create the first cluster member, you cannot add other existing application servers to the cluster. Be careful when adding an existing server because the only way to remove an application server from a cluster is to delete the application server. Consider using the existing server as a template for the cluster members but not as a cluster member. Keeping the original application server out of the cluster allows you to reuse the template if you need to rebuild the configuration.
- f. If you chose to add an existing server, specify the server weight. 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 assigned to other servers in the cluster, then the 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.

### 2. Create cluster members.

**Note:** With WebSphere Application Server Version 6.0, you can upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you might be managing servers that are at the current release and servers that are running the newer release in the same cell. Note that in this mixed environment, there are some restrictions on what you can do with clusters and cluster members:

- You cannot create a cluster member using a server running on WebSphere Application Server Version 5.x.
- You cannot add a member that is running WebSphere Application Server Version 5.x to any cluster.
- You can add a new WebSphere Application Server Version 6.0 cluster member to a mixed cluster only if the cluster already contains a WebSphere Application Server Version 6.0 member that was upgraded from WebSphere Application Server Version 5.x. All new cluster members must be running WebSphere Application Server Version 6.0.

For each new cluster member, perform the following actions:

- a. Type the name of a new application server (cluster member) to add to the cluster.
  - b. Select the node on which the server resides.
  - c. Specify the server weight. The weight value controls the amount of work directed to the application server. If the weight value for the server is greater than the weight values 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. Make sure that **Generate Unique HTTP Port** is selected.
  - e. Specify the server template. You can choose the default application server template or an existing application server template. All the application servers you add after the first application server use the same template.
  - f. Click **Apply** to finish the cluster member. You can add more cluster members. All cluster members you add are based on the same server template.
3. **View the summary.** View a summary of the changes and then click **Finish**. Your cluster is created.
  4. Define a virtual host with a unique port number. See “Configuring virtual hosts” on page 168 for more information.
    - a. In the administrative console, click **Environment > Virtual Hosts > default\_host > Host Aliases**.
    - b. Click **New** and, on the settings page for a virtual host that displays, specify an administrative name for the virtual host.
    - c. Under **Additional Properties**, click **Host Aliases** and define a unique port number for the virtual host.
  5. Save your configuration. As part of saving the change to the configuration, you can select **Synchronize changes with Nodes** before clicking **Save** on the Save page.
  6. Before you can start the cluster, the configuration needs to be synchronized to the nodes. If you selected **Synchronize changes with Nodes** when saving your configuration in the previous step, you can ignore this step. If you are running automatic synchronization, wait until synchronization runs. Or, run manual synchronization to get the configuration files moved to the nodes. Click **System administration > Nodes** and, on the Nodes page, select the node and click **Synchronize** or **Full resynchronize**. The Nodes page displays status indicating whether the node is synchronized.
  7. To further configure a cluster, click **Servers > Clusters**. To view the settings for the cluster, click on the cluster **Name**. Note that unless you have clicked **Save** and saved your administrative configuration, you only see the **Configuration** and **Local Topology** tabs; to see the **Runtime** tab you must save your administrative configuration. Also, ensure that changes are synchronized to the nodes (step 7).

You can create cluster members or start the cluster. See Chapter 12, “Balancing workloads with clusters,” on page 261 for more information about cluster configuration options.

Use scripting to automate the task of creating clusters. See the *Administering applications and their environment* PDF for more information.

## Creating a cluster : Basic cluster settings

Use this page to enter the basic cluster settings.

To view this administrative console page, click **Servers > Clusters > New**.

### Cluster name

Specifies the name of the cluster. The cluster name must be unique within the cell.

### Prefer local

Specifies that the node scoped routing optimization is enabled or disabled. The default is enabled, which means that Enterprise JavaBeans (EJB) requests are routed to the client node when possible. Enabling this setting improves performance because client requests are sent to local enterprise beans.

### Create replication domain for this cluster

Specifies that when the cluster is created, a replication domain is also created with the cluster. The replication service transfers both Java 2 Platform, Enterprise Edition (J2EE) application data and any internal data that is used to maintain the application data among WebSphere Application Server run-time processes in a cluster of application servers.

Create a separate replication domain to use with each component that acts as a consumer of the replication. For example, you can configure one replication domain to use with a session manager and another domain to use with dynamic cache. The replication domain name that is created is identical to the cluster name. To manage the replication domain that is created, click **Environment > Replication domains** in the administrative console.

### Existing server

Specifies whether to create an empty cluster or to add an existing application server to the cluster.

To create a cluster without an existing application server as a cluster member, click **Do not include an existing server in this cluster**.

To add an existing server to the cluster, click **Select an existing server to add to this cluster** and choose the application server to add to the cluster. This application server becomes a template for any additional cluster members that you add to the cluster. When adding servers to a cluster, the only way to remove an application server from a cluster is to delete the application server.

### 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.

## Creating a cluster : Basic cluster member settings

Use this page to enter cluster member settings. You can create a cluster member during or after cluster creation.

You can create a new application server to become a cluster member in two ways:

- Create a cluster member when you create the cluster. In the administrative console, click **Servers > Clusters > New** in the administrative console.



- Create a cluster member after you create the cluster. In the administrative console, click **Servers > Clusters > *cluster\_name* > Cluster Members > New**.

### Member name

Specifies the name of the application server that is created for the cluster.

The member name must be unique within the cell.

### Select node

Specifies the node on which the application server resides.

### Core group

Specifies the core group in which the application server resides. This field is displayed only if you have multiple core groups configured. You can change this value for the first cluster member only.

### 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 template

Specifies an application server template for the new application server.

The new application server contains identical configuration settings to the application server template.

Click **Existing application server** to choose from a list of application server templates. The application server that you choose becomes a template for the cluster member.

The template options are available only for the first cluster member that is created. All cluster members that you create after the first member are identical.

## Creating a cluster : Summary settings

Use this administrative console page to save settings that you modified when creating a cluster or cluster member.

You can view this administrative console page when creating a new cluster or a new cluster member. This summary page displays your configuration changes before you commit the changes and a new cluster or cluster member is created. To create a cluster or cluster member, click the following paths in the administrative console:

- Click **Servers > Clusters > New** to create a new cluster.
- Click **Servers > Clusters > *cluster\_name* > Cluster Members > New** to create new application servers for an existing cluster.

Review the changes to your configuration. Click **Finish** to complete and save your work. The bounding node group of the cluster is based on the first application server that is added as a member of the cluster. To select a different bounding node group, click **Servers > Clusters > *cluster\_name*** in the administrative console to edit the settings for the cluster.

## Server cluster collection

Use this page to view information about and manage clusters of application servers.

To view this administrative console page, click **Servers > Clusters**.

Click **New** to access the Create New Cluster page, which you use to define a new cluster.

## Name

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

## Status

Specifies whether cluster members are stopped, starting, or running.

If all cluster members are stopped, the cluster status and state is *stopped*. After you request to start a cluster by clicking **Start** or **Ripplestart**, 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 *partially started*. The state remains *partially started* until all cluster members are running, then the state changes to *running* and the status is *started*. Similarly, when stopping a cluster by clicking **Stop** or **ImmediateStop**, the state changes to *partially stopped* as the first member stops and changes to *stopped* when all members are not running.

## Server cluster settings

Use this page to view or change the configuration and local topology of a server cluster instance. Provided you saved your administrative configuration after creating the server cluster instance, you can also view run-time information such as the status of the server cluster instance.

To view this administrative console page, click **Servers > Clusters > cluster\_name**.

### **Cluster name:**

Specifies a logical name for the cluster. The name must be unique among clusters within the containing cell.

**Data type** String

### **Bounding node group name:**

Specifies the node group that bounds 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 WebSphere Application Server nodes. A node is a logical grouping of managed servers, usually on a computer 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.

### **Prefer local:**

Specifies whether enterprise bean requests are routed to the node on which the client resides, if it is possible to do so.

Select the **Prefer Local** check box to specify routing of requests to the node on which the client resides. By default, the **Prefer Local** check box is selected, specifying routing of requests to the node.

**Data type** Boolean  
**Default** true

### **Enable high availability for persistent services:**

Specifies that the recovery logs for persistent services reside on storage devices that have been configured according to high availability requirements.

#### **wlclD:**

Specifies the currently registered workload controller (WLC) identifier for the cluster. This setting might not display for all configurations.

**Data type** String

#### **State:**

Specifies whether cluster members are 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.

**Data type** String

**Range** Valid values are starting, partially started, running, partially stopped, or stopped.

---

## **Creating cluster members**

Use this task to create application servers to be members of a configured cluster.

Create a cluster. See “Creating clusters” on page 264 for more information.

You create a cluster member to represent an application server in a cluster. To create a cluster member, view information about cluster members, or manage members of a cluster, use the Cluster Members page.

**Note:** With WebSphere Application Server Version 6.0, you can now upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you might be managing servers that are at the current release and servers that are running the newer release in the same cell. Note that in this mixed environment, there are some restrictions on what you can do with clusters and cluster members:

- You cannot create a cluster member using a server that is running on WebSphere Application Server Version 5.x.
  - You cannot add a member that is running WebSphere Application Server Version 5.x to any cluster.
  - You can add a new WebSphere Application Server Version 6.0 cluster member to a mixed cluster only if the cluster already contains a WebSphere Application Server Version 6.0 member that was upgraded from WebSphere Application Server Version 5x. All new cluster members must be running WebSphere Application Server Version 6.0.
1. Click **Servers > Clusters** in the administrative console. Then, click a cluster in the collection of clusters and click **Cluster members**. The Cluster members page lists members of a cluster, states the nodes on which members reside, and states whether members are started, stopped or encountering problems.
  2. Click **New** and follow the steps on the Create new cluster members page.

- a. Type a name for the cluster member (application server).
  - b. Select the node for the server.
  - c. Specify the server weight. 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 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 whether to generate a unique HTTP port.
  - e. Specify the server template.
  - f. Click **Apply** to finish the cluster member. Repeat these steps to define another cluster member.
  - g. Click **Next**.
  - h. Review the summary of information on new cluster members and click **Finish**.
3. Click **Save** to save your administrative configuration.
  4. To examine a cluster member's settings, click on the member's name under **Member name** on the Cluster members page. This displays the settings page for the cluster member instance.

You created application servers that became members of an existing server cluster.

If you are finished configuring your cluster, you can create a backup cluster and start the cluster. See "Creating backup clusters" on page 271 and "Starting clusters" on page 275 for more information.

You can automate the task of adding cluster members by using scripting. See the *Administering applications and their environment* PDF for more information.

## Cluster member collection

Use this page to view information about and manage members of an application server cluster.

To view this administrative console page, click **Servers > Clusters > cluster\_name > Cluster 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.

### Status

Specifies whether a cluster member is running, stopped, or unavailable.

If a cluster member is stopped, its status is Stopped. After you request to start a cluster member by clicking **Start**, the status becomes Started. After you click **Stop**, its status changes to Stopped when it stops running.

Note that 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 cluster member.

## Cluster member settings

Use this page to configure a member instance of an application server cluster.

To view this administrative console page, click **Servers > Clusters > cluster\_name > Cluster members > cluster\_member\_name**.

### **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.

**Data type** String

**Weight:**

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 assigned to other servers in the cluster, then the server receives a larger share of the server workload.

**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** Integer

---

## Creating backup clusters

Use this task to configure a backup cluster that handles Enterprise JavaBeans (EJB) requests if the primary cluster fails.

Before you begin, create two clusters that are able to provide backup for each other. The objects and resources available in the primary cluster must also be available in the backup cluster. You must use the same cluster name, install the same applications, use the same application names, and define the same resources in the backup cluster as in the primary cluster.

The primary cluster and the backup cluster must reside in separate cells because a cluster must have a unique name within a cell. See “Backup clusters” on page 272 for more information.

Perform this task to create a backup cluster for your EJB clusters. When all the servers in the primary cluster fail, work is not halted because the backup cluster can continue serving requests for EJB work.

To configure a backup cluster, specify a name and a port. The port is called a domain bootstrap address and consists of a bootstrap host and port. The bootstrap host is the host that contains the deployment manager in which the backup cluster is configured. The bootstrap port is equal to the bootstrap port for the same deployment manager.

The primary cluster and the backup cluster must reside in separate cells. The bootstrap host and port for the backup cluster determine which cell contains the backup cluster.

1. Determine the bootstrap host and port of the backup cluster.
  - a. Connect the administrative console for the deployment manager that contains the backup cluster.
  - b. Click **System administration > Deployment manager > Ports > BOOTSTRAP\_ADDRESS**. The host and port for the BOOTSTRAP\_ADDRESS instance is the host and port that the backup cluster uses. Remember these values for when you configure the primary cluster.
2. Connect the administrative console to the deployment manager that contains the primary cluster. Click **Servers > Clusters > cluster\_name > Backup cluster**.
3. Ensure that the name of the backup cluster is the same as the primary cluster.

4. Click **Domain bootstrap address**. Specify the backup cluster deployment manager bootstrap host and port in the **Host** and **Port** fields. Click **OK**. The bootstrap host and port combined define a bootstrap address for the deployment manager. On the Domain Bootstrap Address page, use the **Configuration** tab to statically define the backup cluster; the static value is consumed each time the deployment manager starts. You can use the **Runtime** tab to define the backup cluster during run time only; when the deployment manager stops, the run-time backup cluster information is discarded.
  5. Click **OK**.
  6. Configure a core group bridge between each of the cluster core groups. Use an access point group to join the two core groups. In the deployment manager for the primary cell, configure an access point group that has a peer access point that refers to the core group access point in the backup cell. In the deployment manager for the backup cell, create an access point group that has the same name as the access point group that you created in the primary cell. Add a peer access point that refers to the core group access point in the primary cell. See “Configuring communication between core groups that are in different cells” on page 334 for more information.
- Tip:** If you are configuring a V5.x cluster to back up a cluster that is on the current release, you do not have to configure the core group bridge service because the V5.x cluster does not belong to a core group. The backup cluster still functions using only the domain bootstrap address.
7. Save the configuration.

The backup cluster completes EJB requests when the primary cluster fails.

If you experience problems when configuring your backup cluster, see the *Troubleshooting and support* PDF.

## Backup clusters

Backup clusters mirror the primary server clusters. Mirrored cluster support is for Enterprise JavaBeans (EJB) requests only.

### Overview and prerequisites

When all the members of a cluster are no longer available to service EJB requests, any clients that must interact with one of the EJB application servers in the cluster do not function. Mirrored clusters enable an EJB cluster (primary cluster) to failover to another EJB cluster (backup cluster) when none of the EJB application servers in the primary cluster are able to service a request. The backup cluster allows the client to continue functioning when all of cluster members in the primary cluster are not available.

The fail back is automatic. You do not have to initiate fail back to the primary cluster after restarting the servers in the primary cluster. The backup cluster stops servicing requests as soon as the primary cluster becomes available.

For the backup cluster to take over servicing requests successfully, the objects and resources available in the primary cluster must also be available in the backup cluster. You must use the same cluster name, install the same applications, use the same application names, and define the same resources in the backup cluster as in the primary cluster.

The primary cluster and the backup cluster must reside in separate cells because a cluster must have a unique name within a cell.

For successful fail back, the primary cluster must be defined as the backup for the backup cluster. Both the primary and backup clusters must have a backup cluster configured, and each cluster must specify the opposite cluster as its backup cluster.

Because the primary and backup clusters reside in different cells, with the current version of WebSphere Application Server, the clusters also reside in different core groups. You must configure the core group

bridge service to allow communication between core groups. The core group bridge service eliminates the requirement of a running deployment manager and a node agent for the backup cluster support. In the previous release, if the deployment manager stopped, new requests could not be forwarded to the backup cluster after the primary cluster failed. In WebSphere Application Server V6, any core group bridge server that is configured in the primary cluster's cell can provide information about the backup cluster. The backup cluster support fails only if all of the core group bridge servers in a cell are not running.

For cluster failover and fail back to function as expected, all of the servers (deployment manager, node agents and application servers) for both the primary cluster and the backup cluster must be at a release and level that provides mirrored cluster support; that is, WebSphere Application Server Enterprise or WebSphere Business Integration Server Foundation V5.0.2 or later or WebSphere Application Server Network Deployment V6. However, if you are using a V5.x cluster to back up a cluster that is on the current release, you do not have the additional functionality from the core group bridge service. All the deployment managers must be functional for backup cluster support.

## Configuration

Mirrored cluster support is not configured by default. To use the mirrored cluster support, you must specify backup clusters in your configuration. Each cluster can have only one backup cluster, which must be configured before it is specified as a backup cluster.

To configure a backup cluster in a cluster, you must specify a name and a domain bootstrap address. The bootstrap host is the host that contains the deployment manager in which the backup cluster is configured. The bootstrap port is the bootstrap port for this deployment manager.

The primary cluster and the backup cluster must reside in separate cells. To place mirrored clusters in separate cells, configure the appropriate backup cluster domain bootstrap address. The backup cluster bootstrap host and port determine which cell contains the backup cluster.

You can configure a backup cluster using the administrative console or the ExtendedCluster MBean. To determine the value for the domain bootstrap address and configure a backup cluster using the console, see [Creating backup clusters](#).

To configure a backup cluster using the administrative console, use the **Configuration** tab on the Domain Bootstrap Address page to statically define the backup cluster; the static value is consumed each time the deployment manager starts. Use the **Runtime** tab on the Domain Bootstrap Address page to define the backup cluster during run time only; when the deployment manager stops, the run-time backup cluster information is discarded.

Because the primary and backup clusters reside in different cells, with the current version of WebSphere Application Server, the clusters also reside in different core groups. You must configure the core group bridge service to allow communication between core groups. Use an access point group to join the two core groups. In the deployment manager for the primary cell, configure an access point group that has the core group access point for the backup cell as a peer access point. In the deployment manager for the backup cell, create an access point group that has the same name as the access point group you created in the primary cell. Add a peer access point that refers to the core group access point in the primary cell. See "Configuring communication between core groups that are in different cells" on page 334 for more information. If you are configuring a V5.x cluster to back up a cluster that is on the current release, you do not have to configure the core group bridge service because the V5.x cluster does not belong to a core group. The backup cluster still functions using only the domain bootstrap address.

If you are configuring a backup cluster using the ExtendedCluster MBean, you can change the runtime configuration only. The MBean change does not affect the static configuration. You can use the `setBackup` operation on the ExtendedCluster MBean to change the run-time configuration. For example, you can use the following Java code to set the primary cluster's backup cluster:



```
ac.invoke(extendedCluster, "setBackup", new Object[] {
    backupClusterName, backupBootstrapHost, backupBootstrapPort},
    new String[] {
        "java.lang.String", "java.lang.String", "java.lang.Integer"});
```

In this sample, *ac* is the AdminClient, and *extendedCluster* is the ExtendedClusterObjectName for the primary cluster.

## Fail back support scenarios

There are two scenarios that affect the cluster fail back support.

In the first scenario, requests are made by the client to the primary cluster, which eventually stops accepting requests. The requests are then routed to the backup cluster. The client initially sent requests to the primary cluster and therefore has information about the primary cluster. As a result, when the primary cluster is available again, the requests fail back to the primary cluster.

In the second scenario, the client does not start sending requests until after the primary cluster is down, and the requests go directly to the backup cluster. In this case, the client has information about the backup cluster only. Because the client knows about the backup cluster only, when the primary cluster becomes available, the requests from this client continue to route to the backup cluster and do not fail back to the primary cluster when it becomes available. This scenario occurs when an object is created on the backup cluster. In this case, the backup cluster becomes the primary cluster for this object.

Both of these scenarios can occur within the same client at the same time, if the client is sending requests to existing objects and creating new objects after the primary cluster stops processing.

## Backup cluster settings

Use this page to configure a backup server cluster. The backup server cluster is used if the primary server cluster fails.

Configuration of a backup cluster is only useful if the cluster contains an Enterprise JavaBeans (EJB) module and a client outside of the cluster uses the EJB module.

You can view run-time information such as the status of the backup cluster, provided you saved your administrative configuration after configuring the backup cluster, restarted the server and the workload management configuration reads the backup cluster value when the server starts.

To view this administrative console page, click **Servers > Clusters > *cluster\_name* > Backup cluster**.

### Backup cluster name

Specifies the name of the backup cluster. The backup cluster must have the same name as the server cluster that is containing the backup cluster. The backup cluster and its containing server cluster can have identical names because they must reside in different cells.

**Data type** String

### Domain bootstrap address settings

Use this page to specify the bootstrap address host and port of the deployment manager that contains the backup cluster.

When values is shown for the bootstrap address port, the backup cluster is not set. No host or port values are shown if a user has never set values for the backup cluster or if the user defined a backup cluster with an actual host and port and then removed the backup cluster by removing all text from the host and port fields.

After you set host and port values on the **Configuration** tab, the bootstrap address endpoint values become active when you restart the server. Use the **Runtime** tab to update the bootstrap address host and port dynamically. The system uses the run-time values until you restart the server or change the values. The values on the two tabs are independent. That is, you can specify a run-time backup cluster that is different from the configuration backup cluster.

To view this administrative console page, click **Servers > Clusters > cluster\_name > Backup cluster > Domain bootstrap address**.

**Host:**

Specifies the IP address, domain name server (DNS) host name with domain name suffix, or just the DNS host name, of the bootstrap host for the deployment manager of the backup cluster.

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.

**Data type** String

**Port:**

Specifies the bootstrap port number for the deployment manager of the backup cluster. The port value is used in conjunction with the host name.

**Data type** Integer

---

## Starting clusters

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 244 for more information on how to change the debug port.

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.

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*.

1. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page.
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.

By starting the members of a cluster, you automatically enabled workload management.

**Windows Note to Windows users:** If you start and stop application servers that are part of a cluster using the Windows Services facility, the cluster state does not always update correctly. For example, if a cluster is running and you stop a cluster member through the Services GUI, the cluster state remains as *started* even though the server is no longer running.

See Chapter 12, “Balancing workloads with clusters,” on page 261 for more information about the tasks that you can complete with clusters.

---

## Stopping clusters

Use this task to stop a cluster and any application servers that are members of that cluster.

You can stop all application servers that are members of a cluster at the same time by stopping the cluster. That is, you can stop all application servers in a server cluster at the same time.

**Windows Note to Windows users:** If you start and stop application servers that are part of a cluster using the Windows Services facility, the cluster state does not always update correctly. For example, if a cluster is running and you stop a cluster member through the Services GUI, the cluster state remains as *Started* even though the server is no longer running.

1. Click **Servers > 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*.

See Chapter 12, “Balancing workloads with clusters,” on page 261 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 WebSphere Application Server component that performs replication services, including replicating data, objects, and events among application servers.

If you configured a data replication domain with a previous version of WebSphere Application Server, you might be using a multi-broker replication domain. Any replication domains that you create with the current version of WebSphere Application Server are data replication domains. Migrate any multi-broker replication domains to data replication domains. To learn the differences between the two types of replication domains, see “Comparison of multi-broker versus data replication domains” on page 281 and “Migrating V6.0 servers from multi-broker replication domains to data replication domains” on page 280.

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. For more information about replication, see “Replication” on page 277.

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. See “Data replication domain settings” on page 279 for more information about the properties that you can configure for your replication domain.
  - **Create a replication domain when creating a cluster.**

To create a replication domain when you create a cluster, click **Servers > Clusters > New** in the administrative console. Enable **Create replication domain for this cluster**. 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 > replication\_domain\_name** in the administrative console. See “Creating clusters” on page 264 for more information about creating a cluster.

For more information about the replication domain settings that you can configure in the administrative console, see “Data replication domain settings” on page 279

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 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.
  - To configure dynamic cache replication, see the *Developing and deploying applications* PDF.
  - To configure memory-to-memory replication for session manager, see the *Developing and deploying applications* PDF.
  - To configure replication of stateful session beans, see the *Developing and deploying applications* PDF.

Data is replicating among the application servers in a configured replication domain.

If you use the Data Encryption Standard (DES) or DES3 encryption type for a replication domain, an encryption key is used for the encryption of messages. Click **Regenerate encryption key** on the “Data replication domain settings” on page 279 page at regular intervals to regenerate the key and protect your configuration. After the key has been changed or regenerated, you must restart all of the application servers that are configured as part of the replication domain. Consider performing this step every month to secure your configuration.

## 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 *Developing and deploying applications* 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 *Developing and deploying applications* 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.

By having a single replica configuration defined, you can 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. Any replication domain that is used by dynamic cache must use a full group replica.

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.

To configure replication, see “Replicating data across application servers in a cluster” on page 276.

## 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.

|                |           |
|----------------|-----------|
| <b>Units</b>   | seconds   |
| <b>Default</b> | 5 seconds |

### **Encryption type:**

Specifies the type of encryption to use when transferring replicated data to another area of the network. The options are NONE, DES, and DES3. The default is NONE. The DES and DES3 options encrypt data sent between application server processes (for example, session manager and dynamic caching) to better secure the network joining the processes.

If you specify DES or DES3, a key for global data replication is generated after you click **Apply** or **OK**. If you use the DES or DES3 encryption type, click **Regenerate encryption key** at regular intervals, for example once each month. Periodically regenerating the key enhances security.

|                  |        |
|------------------|--------|
| <b>Data type</b> | String |
| <b>Default</b>   | 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.

|                                    |                                                                                                                   |
|------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>Single replica</b>              | One replica is created. This is the default value.                                                                |
| <b>Full group replica</b>          | Each object is replicated to every application server that is configured as a consumer of the replication domain. |
| <b>Specific number of replicas</b> | A custom number of replicas for any entry that is created in the replication domain.                              |



## Migrating V6.0 servers from multi-broker replication domains to data replication domains

Use this task to migrate multi-broker replication domains to data replication domains. Multi-broker domains were created with a previous version of WebSphere Application Server.

For HTTP session affinity to continue working correctly when migrating V5.x application servers to V6.0 application servers, you must upgrade all of the Web server plug-ins for WebSphere Application Server to the latest version before upgrading the application servers that perform replication.

After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicators with the administrative console.

The different versions of application servers cannot communicate with each other. When migrating your servers to the current version of WebSphere Application Server, keep at least two application servers running on the previous version so that replication remains functional.

Perform this task on any multi-broker domains in your configuration after all of your servers that are using this multi-broker domain have been migrated to the current version of WebSphere Application Server. For more information about the differences between multi-broker domains and the data replication domains, see “Comparison of multi-broker versus data replication domains” on page 281.

The following examples illustrate the migration process for common configurations:

### Migrating an application server configuration that uses an instance of data replication service in peer-to-peer mode

Use this migration path to migrate a replication domain that uses the default peer-to-peer configuration. Dynamic cache replication domains use the peer-to-peer topology. See the *Developing and deploying applications* PDF for more information.

Before you begin, migrate all the Web server plug-ins for your application server cluster to the current version.

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See “Replicating data across application servers in a cluster” on page 276 for more information about creating a data replication domain.
3. Add your migrated application servers to the new data replication domain. For example, if you are migrating 4 servers, migrate 2 servers first and add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. When the new data replication domains are successfully sharing data, migrate the rest of the servers that are using the multi-broker replication domain to data replication domains.
5. Delete the empty multi-broker replication domain.

### Migrating an application server configuration that uses an instance of the data replication service in client/server mode

Use this set of steps to migrate a replication domain that uses 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:

1. Migrate one or more of your existing servers to the current version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See “Replicating data across application servers in a cluster” on page 276 for more information about creating a data replication domain.
3. Add your migrated servers to the new data replication domain. For example, if you are migrating 4 servers, migrate two of these servers and then add them to the new replication domain. Configure the servers to use the new domain by configuring the consumers of the replication domain.
4. Add a part of the clients to the new data replication domain.
5. When the new data replication domains are successfully sharing data, migrate the rest of the clients and servers that are using the multi-broker replication domain to data replication domains.
6. Delete the empty multi-broker replication domain.

### **Migrating a replication domain that uses HTTP session memory-to-memory replication that is overloaded at the application or web module level**

1. Upgrade your deployment manager to the current version of WebSphere Application Server. All the application servers remain configured with the old multi-broker domains on the previous version of WebSphere Application Server.
2. In the administrative console, create an empty data replication domain. Click **Environment > Replication domains > New** in the administrative console. See “Replicating data across application servers in a cluster” on page 276 for more information about creating a data replication domain.
3. Migrate each application server to the current version of WebSphere Application Server, one at a time. The remaining servers on the previous version of WebSphere Application Server can still communicate with each other, but not with the migrated servers. The migrated servers can also communicate with each other.
4. Continue migrating all of the servers to the current version of WebSphere Application Server. All of the application servers are still using multi-broker replication domains, so the features of data replication domains cannot be used.
5. Configure all of the application servers to use the new data replication domain, adding the application servers to the empty replication domain that you created.
6. Restart all of the application servers in the cluster.
7. Delete the empty multi-broker replication domain.

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 session manager and another replication domain for dynamic cache. See “Replicating data across application servers in a cluster” on page 276 for more information.

### **Comparison of multi-broker versus data replication domains**

Data replication domains replace multi-broker domains for data replication between application servers in a cluster.

Any replication domains that are created with a previous version of WebSphere Application Server might be multi-broker domains. Migrate any multi-broker domains to the new data replication domains. Although you can configure existing multi-broker domains with the current version of WebSphere Application Server, after you upgrade your deployment manager, you can create only data replication domains in the administrative console.

Multi-broker and data replication domains both perform the same function, which is to replicate data across the consumers in a replication domain. Configure all the instances of replication that need to communicate

in the same replication domain. You can also configure the session manager with both types of replication domains to use topologies such as peer-to-peer and client/server to isolate the function of creating and storing replicas on separate application servers. You can control the redundancy of replication for each type of replication domain. With a data replication domain, you can specify a specific number of replicas.

If you used multi-broker domains with earlier releases of WebSphere Application Server, use the following comparison chart to learn the differences between how V5.x and V6.0 application servers use the two types of replication domains:

|                          | <b>V5.x application servers using replication domains</b>                                                                                                            | <b>V6.0 application servers using replication domains</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Replication domain types | Uses only multi-broker replication domains for replication.                                                                                                          | Servers that are using the current version of WebSphere Application Server 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 V5.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 WebSphere Application Server. 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 WebSphere Application Server did not support data replication domains. The current version of WebSphere Application Server 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. |

|                                  | <b>V5.x application servers using replication domains</b>                                                                                                                                                                                                                                      | <b>V6.0 application servers using replication domains</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Replication domain configuration | The earlier version of WebSphere Application Server encourages the sharing of replication domains between different consumers, such as session manager and dynamic cache.                                                                                                                      | The current version of WebSphere Application Server 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. |
| 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.                                                                                                |

To migrate multi-broker domains to data replication domains, see “Migrating V6.0 servers from multi-broker replication domains to data replication domains” on page 280.

## Replicating data with a multi-broker replication domain

Use this task to work with replication domains that were created and used with a previous version of WebSphere Application Server.

Multi-broker domains were created with a previous version of WebSphere Application Server, but remain functional in the current version. Although you can configure existing multi-broker domains with the current

version of WebSphere Application Server, 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. See “Migrating V6.0 servers from multi-broker replication domains to data replication domains” on page 280 for more information about the benefit of migrating your replication domains.

If you do not have any existing replication domains, see “Replicating data across application servers in a cluster” on page 276 for information about creating new data replication domains.

If you are performing this task, it is assumed that you configured replication with a previous version of WebSphere Application Server 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 on a multi-broker domain. Specify the values for a particular multi-broker replication domain. The default values are generally sufficient, especially for the pooling and timeout values.
  - 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 WebSphere 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. If you use the DES or TRIPLE\_DES encryption type for a replicator, click **Regenerate encryption key** on the settings for a replication domain instance at regular intervals, such as monthly. Periodically changing the key enhances security.

## Multi-broker replication domains

A multi-broker replication domain is a collection of replicator entry (or *replicator*) instances used by clusters or individual servers within a cell. Multi-broker replication domains were created with a previous release of WebSphere Application Server.

**Note:** After you upgrade your deployment manager to the latest version of WebSphere Application Server, you can create data replication domains only. Any multi-broker domains that you created with a previous release of WebSphere Application Server are still functional, however, you cannot create new multi-broker domains or replicators with the administrative console. See “Comparison of multi-broker versus data replication domains” on page 281 for more information.

A replication entry (or *replicator*) is a run-time component that handles the transfer of internal WebSphere Application Server 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 WebSphere Application Server functions (for example, 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 WebSphere Application Server. 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.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Integer |
| <b>Units</b>     | Seconds |
| <b>Default</b>   | 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 WebSphere 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 a WebSphere Application Server process to a replication domain is transferred to all other WebSphere 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 a WebSphere Application Server process sends data to the replication domain, receives data, or does both. The default is both to receive and send data.

|                  |         |
|------------------|---------|
| <b>Data type</b> | Integer |
| <b>Default</b>   | 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 2, Enterprise Edition (J2EE) 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 WebSphere 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 WebSphere Application Server. When you migrate your deployment manager to the current version of WebSphere Application Server, 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 WebSphere Application Server.

*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 that has cluster members.

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.

1. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page.
2. Make sure the cluster you want to remove is **stopped**. If the cluster is **started**, see “Stopping clusters” on page 276.
3. Remove the cluster. Select the cluster and click **Delete**.
4. Save your configuration. Click **Save** on the administrative console task bar. As part of saving the change to the configuration, select **Synchronize changes with Nodes** before clicking **Save** on the Save page.

The cluster is deleted.

---

## Deleting cluster members

Use this task to remove a cluster member from an existing cluster. Removing a cluster member deletes the associated application server. You cannot remove an application server from a cluster without deleting it.

1. Choose the cluster that contains your cluster member. Click **Servers > Clusters** in the console navigation tree to access the Server Cluster page. Select the cluster for your cluster member and click **Cluster members**.
2. Make sure the cluster member you want to remove is **stopped**. If the cluster is **started**, see “Stopping servers” on page 209.
3. Remove the cluster member from the cluster. Select the cluster member you want and click **Delete**.
4. Save your configuration. Click **Save**. As part of saving the change to the configuration, select **Synchronize changes with Nodes** before clicking **Save** on the Save page.

The cluster member is deleted.

---

## Tuning a workload management configuration

You can set values for several workload management client properties to tune the behavior of the workload management run time. You set the properties as command-line arguments for the Java virtual machine (JVM) process in which the workload management client is running.

**Caution:** Set the values of these properties only in response to problems that you encounter. In most cases, you do not need to change the values. If workload management is functioning correctly, changing the values can produce undesirable results.

To change the property values, you can use the Java Virtual Machine page of the administrative console or use the wsadmin tool. In cases such as where a servlet is a client to an enterprise bean, use the administrative console page for the application server where the servlet is running to configure the properties. The steps below describe how to change the values using the console.

1. Access the Java Virtual Machine page.
  - a. In the administrative console, click **Servers > Application Servers > server\_name > Java and Process Management > Process Definition**.
  - b. On the Process Definition page, click **Java Virtual Machine**.

2. On the Java Virtual Machine page, specify one or more of the following command-line arguments in the **Generic JVM arguments** field:

- **-Dcom.ibm.CORBA.RequestTimeout=*timeout\_interval***

If your application is experiencing problems with timeouts, this argument changes the value for the `com.ibm.CORBA.RequestTimeout` property, which specifies 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.

**CAUTION:**

**Be careful specifying this property; it has no recommended value. Set it only if your application is experiencing problems with timeouts.**

- **-Dcom.ibm.websphere.wlm.unusable.interval=*interval***

If the workload management state of the client is refreshing too soon or too late, this argument changes the value for the `com.ibm.websphere.wlm.unusable.interval` property, which specifies the time interval 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. *interval* is the time in seconds between attempts. 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.

3. Click **OK**.

4. Stop the application server and then restart the application server.

---

## Workload management run-time exceptions

The workload management service might create the following exceptions if it encounters problems:

**org.omg.CORBA.TRANSIENT with a minor code 1229066306 (0x40421042)**

This exception is created if the workload management routing service cannot retry a request and the failure resulted from a connection error. This exception indicates that the application should invoke some compensation logic and resubmit the request.

**org.omg.CORBA.NO\_IMPLEMENT with a minor code 1229066304 (0x49421040)**

This exception is created if the workload management service cannot contact any of the Enterprise JavaBeans (EJB) application servers that participate in workload management.

The WebSphere Application Server client can catch these exceptions and then implement its own strategies to handle the situation. For example, it can display an error message if no servers are available.

The workload management routing service can reroute a failed request to a different target transparently to the application if the application will not be adversely affected by a second attempt. Currently, the only way is to check if the request did not run in whole or part on the previous attempt. When a request runs in whole or in part, an *org.omg.CORBA.TRANSIENT with the minor code 1229066306 (0x49421042)* exception is created to signal that a request can be made again. This informs the application that another target might be available to satisfy the request, but the request could not be failed over transparently to the application. Thus, the application can resubmit the request. The routing service creates an *org.omg.CORBA.NO\_IMPLEMENT with the minor code 1229066304 (0x49421040)* exception if it cannot locate a suitable target for the request. The exception is created, for example, if the cluster is stopped or if the application does not have a path to any of the cluster members.

---

## Clustering and workload management: Resources for learning

Use the following links to find relevant supplemental information about clustering and workload management. 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 Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information about:

- Programming model and decisions
- Programming instructions and examples

**Programming model and decisions**

- IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series at <http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/SG246198.html>.
- IBM WebSphere Application Server V5.0 System Management and Configuration: WebSphere Handbook Series at <http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/SG246195.html>.

**Programming instructions and examples**

- WebSphere Application Server education at [http://www.software.ibm.com/wsdd/education/enablement/curriculum/cur\\_webappsrvadm.html](http://www.software.ibm.com/wsdd/education/enablement/curriculum/cur_webappsrvadm.html).
- Listing of all IBM WebSphere Application Server Redbooks at <http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere>.



---

## Chapter 13. Setting up a high availability environment

Planning ahead for high availability support is important to avoid the risk of a failure without failover coverage. The application server infrastructure that is managed by a high availability manager includes cells and clusters. These components relate closely to core groups, high availability groups, and the policy that controls the high availability infrastructure.

In a high availability environment that is set up properly, a high availability manager can reassess the environment it is managing and accept new components as they are added to the environment. For example, when a Java virtual machine (JVM) is added to the infrastructure, a discovery process begins. During startup the JVM tries to contact the other members of the core group. When it finds another running JVM, it initiates a join process with that JVM that determines whether or not the JVM can join the core group. If the new JVM is accepted as a member of the core group, all of the JVMs, including the new one, log message HMGR0218I . This message is also displayed on the administrative console.

Message HMGR0218I indicates the number of application servers in the core group that are currently online. If this message is not displayed after a JVM starts, either a configuration problem or a communication problem has occurred. To fix this situation, verify that the application server is running on a current configuration, by either using the deployment manager to tell the node agent to synchronize, or use the **syncNode** command to manually perform the synchronization. If the JVM still cannot join the core group, a network configuration problem exists.

The high availability manager is designed to function in all of the supported WebSphere Application Server topologies. However, a high availability-managed environment must comply to the following rules:

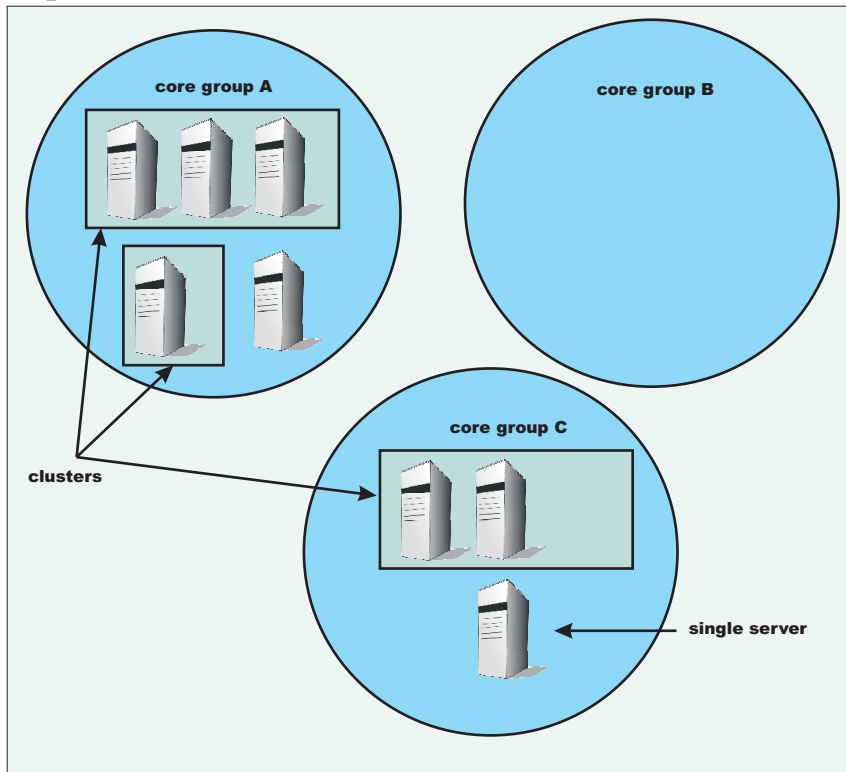
- A cell in a high availability infrastructure is partitioned into one or more core groups. WebSphere Application Server provides a default core group as part of the high availability manager function. Additional core groups can be created using the administrative console.
- A core group cannot extend beyond the boundaries of a cell, and it cannot overlap with any other core groups.
- A cluster must be a member of only one core group. All of the individual members of that cluster must be members of the same core group. This one-to-one relationship between a cluster and a core group exists for both static and dynamic clusters.
- Individual application servers that are part of the high availability environment must also be part of a core group.
- An application server can only join a core group if its JVM can communicate with all of the other online application servers that are part of that core group. If a single application server can not open a connection to the JVM or send a heartbeat to it, the application server is not joined to the core group.

**Tip:** If a communication problem occurs, one of the following situations might exist.

- Two or more of the online application servers are using different IP addresses when they attempt to communicate with the new application server. If the new application server's host name can resolve to multiple IP addresses or if the application server is configured with multiple host files, specify a preferred IP address for the application server.
- One or more application servers in the core group are using a network card or a different subnet or physical network than the other application servers. This situation results in two visible partitions: one partition for each network and subnet. To fix this problem, specify the preferred IP address for each application server and make sure that the IP address for the new application server is on a compatible subnet as well as on the same physical network as the other application servers.

The following diagram illustrates what a cell might look like in a high availability environment:

cell\_1



**Important:** After you set up your WebSphere Application Server environment to comply with all of the high availability-managed environment rules, use the default core group to control this environment. **DO NOT** add additional core groups unless your environment absolutely requires them. Also, do not change the default configurations unless you are doing so to solve a specific problem or situation. When you do make configuration changes, such as changing the policy for a high availability group or moving core group members between core groups in a multi-core group environment, make sure you fully understand the effect such changes will have on your entire environment.

Following are tasks you might perform if you need to change the default configuration:

1. Create additional core groups, if required..
2. Modify the attributes of an existing core group.
3. Modify the attributes of an existing high availability group policy.
4. Create a new policy and associate it with a high availability group.
5. Create core group access points if you create additional core groups.

---

## High availability manager

WebSphere Application Server uses a high availability manager to eliminate single points of failure. A high availability manager is responsible for running key services on available application servers rather than on a dedicated one (such as the deployment manager). It takes advantage of fault tolerant storage technologies such as network attached storage (NAS), which significantly lowers the cost and complexity of high availability configurations. The high availability manager also provides peer-to-peer failover for critical services by always maintaining a backup for these services.



A high availability manager continually monitors the application server environment. If an application server component fails, the high availability manager takes over the in-flight and in-doubt work for the failed server. This action significantly improves application server availability.

In a highly available environment, all single points of failure are eliminated. Because the high availability manager function is dynamic, any configuration changes that you make and save while an application server is running are eventually be picked up and used. You do not have to restart an application server to enable a change. For example, if you change a policy for a messaging engine high availability group while the messaging engine is running, the new policy is dynamically loaded and applied, and the behavior of the messaging engine reflects this change.

A high availability manager focuses on recovery support and scalability in the following areas:

- Messaging
- Transaction managers
- Workload Management (WLM) controllers
- Application servers
- WebSphere partitioning facility instances

To provide this focused failover service, the high availability manager supervises the Java Virtual Machines (JVMs) of the application servers that are core group members. The high availability manager uses one of the following methods to detect failures:

- An application server is marked as failed if the socket fails. This method uses the KEEP\_ALIVE function of TCP/IP, and is very tolerant of extreme application server loading, which might occur if the application server is swapping or thrashing heavily. This method is recommended for determining a JVM failure if you are using multicast emulation, and are running enough JVMs on a single application server to push the application server into extreme CPU starvation or memory starvation.
- A JVM is marked as failed if it stops sending heartbeats for a specified time interval. This method is referred to as *active failure detection*. When it is used, a JVM sends out one heartbeat, or pulse every second. If the JVM is unresponsive for more than 20 seconds, it is considered down. You can use this method with multicast emulation. However, this method must be used for true multicast addressing.

In either case, if a JVM fails, the application server on which it is running is separated from the core group and any services running on that application server are failed over to the surviving core group members.

A JVM can be a node agent, an application server or a deployment manager. If a JVM fails, any singletons running in that JVM are restarted on a peer JVM after the failure is detected. This peer JVM is already running, and eliminates the normal startup time, which potentially can be minutes.

All of the application servers in a cell are defined as members of a core group. Each core group has only one logical high availability manager that services all of the members of that core group. The high availability manager is responsible for making the services within a core group highly available and scalable. It continually polls all of the core group members to verify that they are active and healthy.

A policy matching program is used to localize certain policy-driven components and to place these components into high availability groups. When a core group member fails, the high availability manager assigns the failing member's work to the same type of component from the same high availability group. Using NAS devices in the position of common logging facilities helps to recover in-doubt and in-flight work if a component fails.

WebSphere Application Server provides a default core group that is created during installation. New server instances are added to the default core group as they are created. The WebSphere Application Server environment can support multiple core groups, but one core group is usually sufficient for most environments.

A high availability manager is comprised of a variety of components. All of the components in a high availability manager infrastructure work together to ensure peer-to-peer failover is effectively protecting the application server environment from failures. The following table describes the main high availability components, or areas of components, required for an effective high availability manager environment:

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Server component areas   | Focus on the application server run time and include such entities as cells and clusters. These areas are necessary for a healthy high availability manager run time because they closely relate to core groups, high availability groups, and the policy that defines the infrastructure.                                                                                                                                                                                         |
| Core groups              | Provide failover support. A default core group is created during startup. This core group should be sufficient for most environments. Additional core groups can be created, but you should only create them if you fully understand the implications to your high availability environment.<br><br>Core groups are static in nature. The configuration applied to a core group through user-defined policies determines the dynamic relationship within high availability groups. |
| High availability groups | Closely bound to policy definitions. High availability groups are dynamic in nature and are not configured directly by users. Policy match criteria determines the high availability group to which a core group member belongs.                                                                                                                                                                                                                                                   |
| Network components       | Provide the underlying network infrastructure that is crucial to the success of the high availability manager. By default, WebSphere Application Server uses a channel framework protocol. However, a unicast or multicast protocol can also be used. The network components include a technology that enables communication throughout the high availability manager infrastructure.                                                                                              |

All of these components must be active and properly configured to achieve a highly available infrastructure.

---

## High availability network components

The foundation for a highly available environment is dependent on the network that is created for this environment. The WebSphere Application Server high availability manager, by default, uses a channel framework network protocol model to establish network connections. This model enables network ports to be shared among all of the channels within a transport chain. A unicast protocol, which supports communication between a specific receiver and a specific sender, or a multicast protocol, which supports open communication to all receivers, can also be used.

A high availability environment ties all of the network components together. It provides the basis for providing peer-to-peer failover. The moment a component loses its network connection to the rest of the infrastructure, the high availability manager assumes a failure has occurred and assumes the work assigned to that component.

Thousands of high availability groups can exist within a core group. The policies that are associated with a high availability group control how and when members of that group are activated. A group services mechanism is used to communicate high availability group membership information between members of a core group. This information includes the group services that are available and the lightweight component groups that are part of a high availability group.

The group services mechanism also supports the following types of messaging between core group members.

- High speed First In First Out (FIFO) messaging between members of the lightweight component groups.
- High speed view synchronous messaging.
- Java Virtual Machine (JVM) heartbeats or pulses, which are used to indicate that the associated application server is still healthy.

---

## Transport protocol for a high availability manager

The high availability manager network components can be built on either a channel framework, unicast or multicast transport protocol. Channel framework is the default protocol, but there are options and features available with multicast that might be a better match for your application server environment. Multicast emulation can be used with any of these protocols.

The following network protocol configurations can be used with WebSphere Application Server:

### Channel framework

Channel framework is the default network protocol configuration for the high availability manager. It provides a common model for connection management, thread usage, channel management, and message access within WebSphere Application Server. It extends the concept of a networking protocol stack, or transport chain, to the WebSphere run time. Each transport chain consists of one or more types of channels, and each channel 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.

The transport chain configuration settings determine which I/O protocols are supported for that chain. Custom channels that support requirements unique to a particular customer or environment can also be added to a transport chain.

### Unicast protocol

Unicast protocol is a direct method of sending and receiving messages. JVMs are discovered when a new application server attempts to open connections to application servers in the same core group that are already running. The use of TCP/IP makes this suitable for high speed Wide Area Networks (WANs) and Local Area Networks (LANs). This protocol uses less CPU and memory per connection than multicast protocol, and might be more appropriate for larger core groups.

### Multicast protocol

Multicast protocol requires a tuned environment. Heartbeats must be used as the failure detection method with this protocol. In this environment the JVMs cannot be swapped or kept from running. A non responsive JVM is viewed as a failed unit and all work items are taken away from that application server and dispersed among other core group members. The network used by all of the members of the core group must be able to use multicast protocol.

Performance in terms of data rate is about the same for all three protocols. WebSphere Application Server usually performs the best if a channel framework model is used, because this model supports using multiple protocols at the same time. However if you are choosing between a unicast and a multicast protocol, the unicast protocol is usually better because the most common WebSphere Application Server scenario is low fan out or point-to-point messaging. Using the multicast protocol in a low fan out environment forces all the recipients to process the message, even though only a low number of application servers use the message.

---

## Creating a new core group

Before creating a new core group, you must determine which application server processes to add to the core group. For example, if you are creating a new core group because a firewall is used to separate your proxy environment from your application server environment, you might leave the deployment manager, the node agent, and some of the other Application Server processes in the default core group, and move the server processes that exist on the other side of a firewall to the new core group after you create it.

You might want to perform this task if:

- A firewall is used to separate your proxy environment from your application server environment. (Additional core groups are required to provide proper failover support in this topology.)
- The number of open TCP/IP sockets becomes unacceptable. (This could occur if you are using a multicast protocol.)

Every WebSphere process is initially a member of the default core group provided with the WebSphere Application Server product. Using the default core group is sufficient in most configurations.

To create a new core group:

1. In the administrative console, click **Servers > Core groups > Core group settings > New** .
2. In the Name field, specify a unique name for the new core group. This field can only be edited when you create the new core group. Make sure that the name is meaningful and consistent with the names of the other core groups in the cell. It is helpful if the name conveys why the application servers are being moved to this core group. For example, if your company's human resources applications are installed on the application servers that are moved to this core group, you might include HR as part of the core group name.
3. Add a description of this core group that helps other administrators understand the purpose of this core group.
4. In the Number of coordinators field, specify the number of active application servers you want serving as coordinators for this core group. Each core group coordinator can manage up to 10,000 core group components. Specify a value for this field based on the anticipated number of core group components.
5. Select the type of transport that the application servers contained in this core group are using.
  - a. If you select **CHANNEL\_FRAMEWORK**, specify the name of an already defined transport chain in the Channel chain name field. Any value specified for the Multicast port, Multicast group IP start or Multicast group IP end fields are ignored.
  - b. If you select **MULTICAST**, you must enter a port number in the Multicast port field, enter the first IP address in the range of IP addresses that can be used in the Multicast group IP start field, and enter the last IP address in the range of IP addresses that can be used in the field.
  - c. If you select **UNICAST**, any value specified for the Channel chain name, Multicast port, Multicast group IP start or Multicast group IP end fields are ignored.
6. Click **Apply**.
7. On the administrative console panel click **Core groups > core\_group\_name > Policies > New** , and then select the policy that you want to associate with a high availability group in this core group. The high availability groups that are part of the new core group determine the policies that are required for this core group. New policies do not have to be defined if:
  - The processes contained in the new core group do not contain any high availability groups.
  - The new core group only contains processes, such as the service integration bus, for which default policies are provided as part of the high availability manager function.

If you need to define a new policy, the policy options are:

- All active policy: Under this policy, all of the group members are activated.

- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, the active members of a group are statically assigned.

Multiple policies can be defined if different high availability groups require different policies. However, only one policy can be associated with a given high availability group. See for more information about these policies.

**Note:** If you are setting up a policy for a transaction manager, you must select One of N as your policy type because a transaction manager requires that only one server can have access to a failed server's recovery log at any point in time.

8. Click **Next**.
9. Specify a name for the policy that is unique within the scope of the core group.
10. Change the value specified in the Is alive timer field if the default value is too long or too short a time period. This value, specified in seconds, determines how frequently the high availability manager checks the health of the core group members. Valid values are any integer between -1 and 600, inclusive. If -1 is specified, the timer is disabled. If 0 (zero) is specified, the default value of 120 seconds is used.
11. Select **Quorum** if you want to enable quorum checking for the core group. Quorum is a mechanism that can be used to protect resources shared across members of the group in the event of a failure.

**CAUTION:**

**Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.**

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.
- **GroupName.WAS\_CLUSTER=clustername** must be specified as a property in the group name of any high availability group matching this policy.

When enabled, any group that is using this policy does not achieve quorum until a majority of the members are running. For example, if there are *n* members in the group,  $(n/2) + 1$  servers must be online in order to achieve quorum. No group members are activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

12. Click **Apply**, and then select **Match criteria**.
13. On the next panel, click **New** and then specify the match criterion for this policy. A match criterion is a set of one or more name=value pairs of data that can be matched to attributes contained in the name of a high availability group.
  - a. In the Name field, enter the name of one of the properties contained in a high availability group's name that you want to use to associate this policy with that high availability group.
  - b. In the Value field, enter the value of this property that, along with the property name, forms the name=value pair.
  - c. Optional: Add a description of this match criteria in the Description field.
  - d. Click **Ok**.
  - e. If you need to specify additional name=value pairs for your match criterion, repeat this step. Each name=value pair must be entered separately. Repeat this step until you have specified all of the name=value pairs required for this match criterion.
14. Click **Save**, select **Synchronize changes with nodes** and then click **Save** again to save your changes.

You are now ready to add members to this new core group.

## Core groups

A core group is a set of application servers that can be divided up into various high availability groups. It is a statically defined component of the WebSphere Application Server high availability manager function that monitors the application server environment and provides peer to peer failover of application server components.

A core group can contain one or more high availability groups. However, a high availability group must be totally contained within a single core group. Any component, such as the service integration bus component of IBM service integration technologies or the WebSphere Application Server transaction manager, can create a high availability group for that component's use. For example, the service integration bus might need a high availability group to support its messaging engines, and the transaction manager component might need a high availability group to support its transaction logs.

A cell must have at least one core group. The WebSphere Application Server creates a default core group, called DefaultCoreGroup, for each cell. All the server processes and Java virtual machines (JVMs), including node agents on all managed nodes, the deployment manager, and application servers residing within a cell are initially members of this default core group.

When properly configured, the default core group is sufficient for establishing a high availability environment. However, certain topologies require the use of multiple core groups. For example, if a firewall is used to separate the proxy environment from the server environment, an additional core group is required in order to provide proper failover support. For this particular type of environment, application servers outside of the firewall are members of a separate core group from the application servers that are inside of the firewall.

The core group contains a bridge service, which supports cluster services that span multiple core groups. Core groups are connected by access point groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports. It is through this set of bridge interfaces that the core group bridge provides access to a core group.

If you create additional core groups, when you move core group members to the new core groups, remember that:

- Each server process and Java virtual machine (JVM), including node agents on all managed nodes, the deployment manager, and application servers within a cell can only be a member of one core group. Core groups cannot overlap each other.
- If a cluster is defined for the cell, all of the cluster members must belong to the same core group.

Network communication between all the members of a core group is essential. The network environment must consist of a fast local area network (LAN) with full Internet Protocol (IP) visibility and bidirectional communication between all core group members. IP visibility means that each member is entirely receptive to the communications of any other core group member.

A core group consists of the following elements:

### Coordinator

The coordinator is the elected, or default, high availability manager. The coordinator is responsible for tracking all of the members of a core group when members leave, join, or fail. The coordinator is not a single point of failure. In the event of a failure involving the coordinator, the preferred coordinator, or a default, picks up the high availability manager work, including the management of the core group.

### Core group members

Any application server that is created within a cell that is defined as part of a high availability environment is automatically designated as a core group member.



## Messaging subsystem

A messaging subsystem is a subsystem, such as the service integration bus component of IBM service integration technologies, that provides high-speed connections between core group members. This subsystem enables all core group members to quickly and effectively communicate with each other.

## Core group bridge service

The core group bridge service is only utilized in high availability environments that contain multiple core groups. Use the bridge service to provide quick and effective communication between core groups.

**Policy** A policy is used to control which members of a high availability group are activated. Even though a policy is defined at the core group level, it does not apply to the core group. The same policy can be used by several different high availability groups but all of the high availability groups to which it applies must be part of the same core group. A policy is established for a high availability group when that group is created. The following policies can be specified for a high availability group:

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy, *M* group members are activated. The number represented by *M* is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

In a run-time server environment each core group functions as an independent unit. A Java Virtual Machine (JVM) that is contained within a cell can be a member of one core group only. This JVM can be a node agent, an application server or a deployment manager. However, even though the deployment manager can belong to one core group only, it is still responsible for configuring all of the application servers within a cell, even if multiple core groups are defined for that cell.

Within a core group, Java Management Extensions (JMX) connectors are given access to run-time MBeans in one of two ways:

1. They can be attached directly to the application server that contains the run-time MBeans.
2. They can be attached to the deployment manager of that cell.

Attaching the JMX connectors to the deployment manager is the easier approach because you need to specify only the host name and port numbers. However, the deployment manager and the node agent must both be running for the JMX connectors to access the run-time MBeans.

Attaching the JMX connectors directly to the application server that contains the run-time MBeans is a more reliable approach because even if the deployment manager is down, the JMX connectors can still access the run-time MBeans as long as the application server that contains the MBeans is still running, .

The coordinator retains and tracks membership and state changes for a core group. To work efficiently, the coordinator must have enough memory to contain the group and state information for the online members of the core group. Configuring multiple coordinators enables this task to be shared equally among a set of online coordinators.

Using the administrative console, you can designate specific servers as preferred coordinator servers. High availability manager coordinators run in these servers. If no preferred coordinator servers are specified, the high availability manager selects them. To minimize churn on the core group, it is recommended that you designate specific servers as preferred coordinator servers and limit how often you restart these servers. The heap sizes for the JVMs and the relative amount of memory that the JVMs use to host coordinator services need to be tuned to ensure that enough memory is available to hold the group and state information for the core group.



While core group members are statically configured, high availability group members are dynamically selected and only exist as run-time entities. The WebSphere Application Server runtime uses a policy matching program to determine the policy that should be used for each high availability group. Each policy includes a match criterion that consists of a set of name=value pairs. The policy matching program matches these name=value pairs with attributes contained in the name of a high availability group. When a match is made, the policy is associated with that high availability group.

## High availability groups

High availability groups are dynamically created components of a core group. They cannot be configured directly but are directly affected by static data, such as policy configurations, which are specified at the core group level.

A high availability group cannot extend beyond the boundaries of a core group. However, members of a high availability group can also be members of other high availability groups as long as all of these high availability groups are defined within the same core group.

Any WebSphere Application Server component, such as the transaction manager, can create a high availability group for that component to use. The component code must specify the attributes that are used to create the name of the high availability group for that component. For example, to establish a high availability group for the transaction manager:

- Code included in the transaction manager component code specifies the attribute `type=WAS_TRANSACTION`s as part of the name of the high availability group associated with this component.
- The high availability manager function includes the default policy Clustered TM Policy that includes `type=WAS_TRANSACTION`s as part of its match criteria.

Whenever transaction manager code creates a high availability group, the high availability manager matches the match criteria of the Clustered TM Policy to the high availability group member name. In this example, the string `type=WAS_TRANSACTION`s included in the high availability group name is matched to the same string in the policy match criteria for the Clustered TM Policy. This match associates the Clustered TM Policy with the high availability group the transaction manager component creates.

After a policy is established for a high availability group, you can change some of the policy attributes, such as quorum, fail back, and preferred servers. However, you can not change the policy type. If you need to change the policy type, you must create a new policy and then use the match criteria to associate it with the appropriate group.

If you want to use the same match criteria, you must delete the old policy before defining the new policy. You cannot use the same match criteria for two different policies.

**Important:** If the old policy is one of the default policies that IBM provides, it is recommended that you do not delete the old policy. Instead, you should use a different match criteria for your new policy. This new match criteria should include more matches with the attributes contained in high availability group's name than the default policy uses for its match criterion. The policy with the greatest number of matches is the one that is used. Not deleting the IBM provided policy enables you to revert back to that policy if a problem occurs when you use your newly created policy.

Before changing the policy type for a high availability group you must fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus uses a One of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.

## Core group collection

A core group is a component of the high availability manager function. A default core group, called **DefaultCoreGroup** is created for each cell in the WebSphere Application Server environment. A core group can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

To view this administrative console page, click **Servers > Core Groups > Core group settings** .

|                              |                                                                                                                                    |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>                  | Specifies the name of the core group. Click the core group name to edit the settings for that core group. This field is read-only. |
| <b>Description</b>           | Describes the core group. This field is read-only.                                                                                 |
| <b>Connected core groups</b> | Specifies the core groups that are connected to this core group by access points. This field is read-only.                         |

Click **New** to define a new core group. After a core group is defined, several fields become read-only. To change those fields, delete and redefine the core group.

Click **Delete** to delete a core group. A core group must be empty before it can be deleted.

## Core group settings

Use this page to create a core group or to edit an existing core group. A core group is a component of the high availability manager function. It can contain standalone servers, cluster members, node agents and the deployment manager. A core group must contain at least one node agent or the deployment manager.

Before you create a core group you must understand the relationship of core groups in a high availability environment and know how you intend to use each core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *Select an existing core group for editing.*

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

After you specify your core group settings, click **Apply** before defining additional properties or setting up a core group bridge.

Extended information about the core group fields:

### Configuration Tab:

#### Name

This field can only be edited when you create new core groups.

If you are defining a new core group, specify a name that is unique name among the existing core groups. It is helpful to other WebSphere Application Server administrators if the name helps define the use of this core group and if it is consistent with the names of the other core groups in the cell.

This field can contain alpha and numeric characters. The following characters cannot be used in this field:

# \ / , : ; " \* ? < > | = + & % ' "

Also, the name cannot begin with a period (.) or a blank space. A blank space does not generate an error. However, leading and trailing blank spaces are automatically deleted.

For example, DefaultCoreGroup is the name of the core group that contains the deployment manager server process.

## Description

Use this optional field to include a description of the core group. In environments where there are multiple system administrators this field can help these administrators understand the overall organization of the core groups. The supported length of this field is quite large. However, long descriptions take time to load and can cause a delay when displaying the page.

**Example:** "Default Core Group. The default core group cannot be deleted." is the description of the DefaultCoreGroup.

## Number of coordinators

The coordinator is a component in each server in a core group that provides the service functionality of the high availability manager. The coordinator for a core group determines membership and communicates state and status to the other members of the core group.

The default value is one coordinator, although multiple coordinators are advisable for large core groups. All of the group data must fit in the memory of the allocated coordinators. One coordinator can run out of memory in a system with a large core group, which can cause the system to work improperly.

## Transport type

The transport type is a required field that specifies the type of network communication your core group uses to communicate to members and to other core groups. The following transport options are available:

### CHANNEL\_FRAMEWORK

CHANNEL\_FRAMEWORK is the default transport type. It uses the channel framework topology to incorporate port reusability and shared port technology into the communication system.

### UNICAST

UNICAST is a targeted network model that focuses on a direct recipient for communication. This type of communication is most suitable when the intended message is sent to a specific set of recipients.

### MULTICAST

MULTICAST consists of a broadcast network model. This model broadcasts communication across the defined network, depending upon the values that are provided for the multicast settings. Multicast settings are suitable when there are many recipients for the intended message; otherwise broadcast communication tends to overload the network with traffic, and can impact performance goals.

## Channel chain name

Specifies the name of the channel chain if CHANNEL\_FRAMEWORK is selected for transport type.

## MULTICAST settings

Specifies the following settings if a multicast transport type is used:

- **Multicast port**

The port setting tells the coordinator where to scan for transmissions. When setting this value, verify that you are specifying a port that is not used by another network communication device. Setting a port value that has conflicts causes problems with your high availability manager infrastructure.

- **Multicast group IP start**

Specify the starting Internet Protocol (IP) address of the intended communication area.

- **Multicast group IP end**

Specify the ending IP address of the intended communication area. Plan the network to accommodate scalability.

## Additional properties

### Core group servers

Use to display the server processes that belong to the core group. Server processes include the

deployment manager, node agents, application servers, and cluster members. You can use the panel that displays to move server processes to a different core group.

### **Custom properties**

Use to define custom properties to be used for configuration purposes.

### **Policies**

Use to define the policies that the coordinator servers use to select the active members of a core group. You can select from existing policies, or create new custom policies.

### **Preferred Coordinator Servers**

Use to designate core group servers as preferred coordinator servers.

## **Related topics**

### **Core group bridge**

Click on to specify core group bridge communication settings between core groups.

## **Runtime Tab:**

### **Group name properties**

Specify one or more name=value pairs as the match criterion for a high availability group. If you specify more than one name=value pair, use a comma to separate the pairs. You can specify an asterisk (\*) to obtain the selected information for all of the high availability groups within this core group.

When a WebSphere Application Server component creates a new high availability group, it establishes a map of that group's properties as the group name. This map is used to uniquely identify that high availability group.

After you specify a match criterion or an asterisk:

- Select **Calculate** to determine how many high availability groups have names that match this match criterion.
- Select **Show groups** to view a list of the high availability groups that match this match criterion. For each group, this list indicates:
  - Its high availability group name
  - Whether or not quorum has been enabled
  - The policy that is associated with the high availability group. If more than one policy is listed for a high availability group, you must change the match criterion for one or more of your policies so that only one policy is associated with this high availability group.
  - Its status (either the OK icon or the Error icon). If only one policy is listed in the Policy column, the OK icon is displayed in the Status column. If more than one policy is listed Policy column, the Error icon is displayed in the Status column.
- Select **Show servers** to view a list of servers which are hosting active members of the high availability groups that match the specified Group name properties. For each server, this list indicates:
  - The names of the servers which are hosting the active high availability group members.
  - The name of the node on which these servers resides.
  - The version of the WebSphere Application Server product on which these servers are running.
  - The number of high availability group members that are currently active on these servers.

**Example:** Suppose the following high availability groups are defined for a core group:

- Component A uses the following properties for its group name: [ name=compA, policy=oneofN, owner=smith ]
- Component B uses the following properties for its group name: [ name=compB, policy=MofN, owner=smith ]
- Component C uses the following properties for its group name: [ name=compC, policy=oneofN, owner=smith ]

If you specify `policy=oneofN` in the **Group name properties** field and then select **Show groups**, the groups for components A and C are listed.

If you specify `owner=smith` in the **Group name properties** field and then select **Show groups**, the groups for components A, B and C are listed.

If you specify all of component C's name properties in the **Group name properties** field:

```
name=compC,policy=oneofN,owner=smith
```

Then select **Show groups**, only the group for component C is listed. Note that the properties are separated by commas. There are no blank spaces.

---

## Changing a core group's configuration

You might want to perform this task if you need to:

- Change the number of coordinators that are to be active.
- Change the transport type for the core group members.
- Set up some custom properties for your specific environment.
- Add or remove application servers from the list of preferred coordinator servers.

To change a core group's configuration:

1. In the administrative console, click **Servers > Core groups > Core group settings > *core\_group\_name***.
2. If you want to change the number of coordinators that must be active for this core group, specify a new value in the **Number of coordinators** field.
3. If you want to change the transport specification for this core group, in the **Transport type** pull-down, select the type of transport the application servers in this core group are going to use.
  - a. If you select **CHANNEL\_FRAMEWORK**, specify the name of the channel chain in the **Channel chain name** field.
  - b. If you select **MULTICAST**, enter the multicast port number in the **Multicast port** field, enter the first IP address in the range of IP addresses that can be used in the **Multicast group IP start** field, and enter the last IP address in the range of IP addresses that can be used in the **Multicast group IP end** field.
4. If you need to set up a custom property for your specific environment, click **Custom properties > New** and then specify the name and value pair of your custom property.
5. If you need to change the list of preferred coordinator servers, click **Preferred coordinator servers** and then add or delete the appropriate application servers from the list of preferred coordinator servers.
6. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

## Core group and core group bridge custom properties

Use these custom properties for advanced configurations with core groups and core groups that communicate with the core group bridge.

### FW\_PASSIVE\_MEMBER

Use this property in a core group bridge configuration when there is a firewall between the core groups and the secure side of the firewall is configured to listen only.

Set the `FW_PASSIVE_MEMBER` custom property to make the bridge interfaces that are in the core group access point passive. Set the value on the core group access point that is on the secure side of the firewall so that the core group bridge interfaces listen for connections from the unsecured side of the

firewall but do not initiate any connections. The servers on the secure side of the firewall are passive. The custom property should correspond to your defined firewall rules that allow connections from the unsecured region to the secure region only.

To configure this custom property, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Core group access points > core\_group\_access\_point\_name > Show detail > Custom properties > New** in the administrative console.

You also must set this custom property in any peer access points that refer to the core group access points that you configure with this custom property.

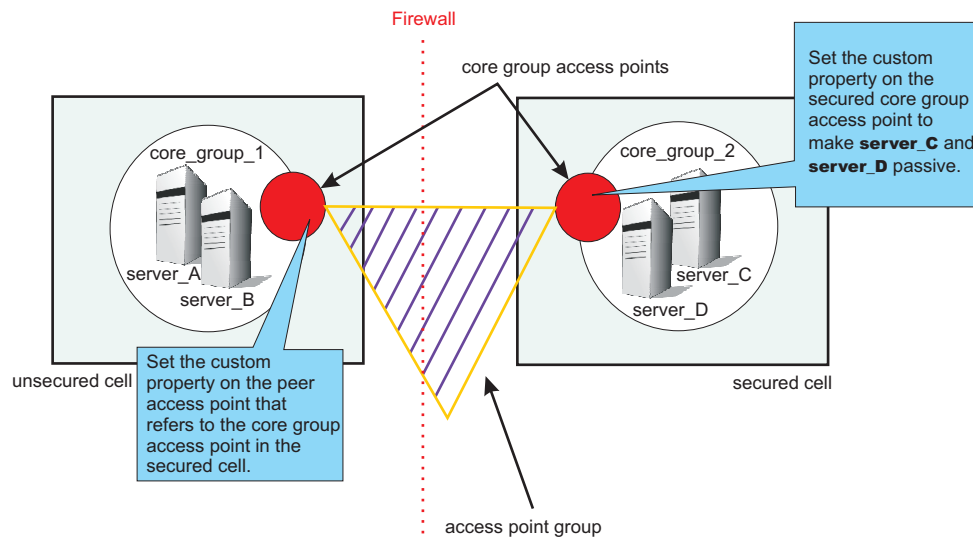


Figure 1. Configuring the FW\_PASSIVE\_MEMBER custom property

For example, *server\_A* and *server\_B* are configured in *core\_group\_1*. *Server\_C* and *server\_D* are configured in *core\_group\_2*. *Core\_group\_2* is behind a firewall that is configured to listen only through the firewall. *Core\_group\_1* and *core\_group\_2* can communicate with each other through an access point group. To configure *server\_C* and *server\_D* to be passive, perform the following steps:

1. In the administrative console for the cell that contains *core\_group\_2*, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Core group access points > core\_group\_access\_point\_name > Show detail > Custom properties > New** .
2. Add the FW\_PASSIVE\_MEMBER custom property. Enter any value to enable the property.
3. In the administrative console for the cell that contains *core\_group\_1*, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Peer access points > peer\_access\_point\_name > Show detail > Custom properties > New**. The peer access point you select should correspond to the core group access point for *core\_group\_2*.
4. Add the FW\_PASSIVE\_MEMBER custom property. Enter any value to enable the property.

By configuring the FW\_PASSIVE\_MEMBER custom property, you configured the servers on the secured side of the firewall, *server\_C* and *server\_D*, to be passive. These servers listen for connections from the other side of the firewall but do not initiate any connections to the unsecured side of the firewall.

## IBM\_CS\_LS\_DATASTACK\_MEG

Use this custom property to eliminate a condition that is reported by a message that is displayed repeatedly in your SystemOut.log file.



You might see a message similar to the following message in the SystemOut.log file multiple times:

```
[9/24/04 13:28:19:497 CDT] 00000013 VSync          W
DCSV2005W: DCS Stack DefaultAccessPointGroup.P at Member 172.16.1.86:9353:
The amount of memory available for synchronization is low. The configured memory
size is 418816 bytes. Currently used memory size is 420307 bytes.
```

If the member IP address is in the format of a dotted decimal IP address and port, you can eliminate these messages by increasing the amount of memory that is allocated to the core stack that is used for core group communication. Increase the value of this property until you no longer see the message in your SystemOut.log file. Because the memory is dynamically allocated, setting a larger stack size than you need does not cause memory problems.

Set the custom property on the bridge interface that contains the particular member that is in the messages. You can also set the custom property on the access point group or the core group access point. If you set the value on the access point group or core group access point, all the bridge interfaces that are in the particular group are affected. If you set the value on an individual bridge interface and an access point group or core group access point, the value that is set for the bridge interface is used. If the value is set on both an access point group and a core group access point, the value that is set for the core group access point is used.

To configure this custom property, complete the following steps:

1. Set the custom property in the administrative console.
  - To set the custom property on a bridge interface, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *core\_group\_access\_point\_name* > Show detail > Bridge interfaces > *bridge\_interface\_name* > Custom properties > New.**
  - To set the custom property on a core group access point, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *core\_group\_access\_point\_name* > Show detail > Custom properties > New.**
  - To set the custom property on an access point group, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Custom properties > New.**
2. Add the IBM\_CS\_LS\_DATASTACK\_MEG custom property. Enter a value that is greater than the default value of 5 megabytes.

|         |           |
|---------|-----------|
| Units   | megabytes |
| Default | 5         |

---

## Changing the configuration of a high availability group

High availability groups are dynamically created components of a core group. They cannot be configured directly but are directly affected by static data, such as policy configurations, which are specified at the core group level.

A high availability group has been established for a core group.

You might want to perform this task if you want to remove a high availability group from a cell.

You can use the administrative console to activate or disable all of the members of a high availability group.

1. In the administrative console, click **Servers > Core groups > Core group settings.**
2. Click on the core group whose high availability groups you want to view.
3. Click on the Runtime tab



4. Specify a match criterion in the Group name properties field that matches the high availability group you want to enable or disable. (You can specify an asterisk (\*) to get a list of all the high availability groups that are part of this core group.)
5. Click show groups.
6. The Status column of the **High availability groups** panel shows the state of the high availability groups in this core group that match the specified match criterion.
7. In the Select column, indicate the high availability group whose state you want to change and then click **Enable** or **Disable**.

**Enable** activates members of the high availability group that are currently in the disabled state. If all of the members are already active, no action is taken.

**Disable** disables all of the high availability group members that are active.

It can take several minutes for this change to take affect.

## High availability group servers collection

Use this page to determine how many high availability group members are active on a particular application server.

To view this administrative console page, click **Servers > Core groups > Core group settings > core\_group**. Click on the **Runtime** tab and specify group name properties for a high availability group. (You can specify an asterisk (\*) to get a list of the servers that are hosting active members for all the high availability groups in this core group.) Then select **Show servers**.

|                       |                                                                                                                            |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>Server</b>         | Specifies the name of a server on which there are active high availability group members. This field is read-only.         |
| <b>Node</b>           | Specifies the node on which each server is running. This field is read-only.                                               |
| <b>Version</b>        | Specifies the version of the WebSphere Application product on which each node is running. This field is read-only.         |
| <b>Active members</b> | Specifies the number of high availability group members that are currently active on that server. This field is read-only. |

## High availability group settings

Use this page to manage the state of a high availability groups.

To view this administrative console page, click **Servers > Core groups > Core group settings > core\_group**. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (\*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups**.

|                                |                                                                                                                                                                                                                                                                                                   |
|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>High availability group</b> | Specifies the names of the high availability groups. The name of a high availability group is a set of name=value pairs or attributes, separated with commas. For example, name=productiongroup,policy=abc,ibm=websphere could be the name of a high availability group. This field is read-only. |
| <b>Quorum</b>                  | Specifies if quorum is enabled for each high availability group. This field is read-only.                                                                                                                                                                                                         |

## Policy

Lists the policies that have match criteria that matches properties contained in the name of that high availability group. There should only be one policy listed for a high availability group. However, if multiple policies have match criteria that equally match properties in a high availability group's name, all of the policies with matching criteria are listed, and the ERROR icon appears in the status column.

For example, if you have a high availability group named `name=productiongroup,policy=abc,ibm=websphere`, and MyPolicy1 has the match criteria `name=productiongroup`, and MyPolicy2 has the match criteria `policy=abc`, both MyPolicy1 and MyPolicy2 are considered matching policies and are listed in the Policy column.

## Status

This field is read-only.

Indicates, with icons, whether or not only one policy is associated with a high availability group. If the OK icon displays in this column, a single policy is associated with that high availability group. If the ERROR icon displays in this column, multiple policies are associated with that group.

If the ERROR icon displays for a high availability group, you must adjust the match criteria for one or more of the policies listed in the Policy column for that group so that the correct policy is the only one associated with that high availability group.

The match criteria for multiple policies can match some of the same properties in a group's name as long as one policy has a match criteria that matches more of the properties in that group's name than the match criteria of any of the other policies. For example, if you have a high availability group with a name that consists of the following name and value pairs:

```
name=productiongroup,policy=abc,ibm=websphere
```

and MyPolicy1 has the match criteria `name=productiongroup` and MyPolicy2 has the match criteria `name=productiongroup,ibm=websphere` MyPolicy2 is considered the matching policy because it has more match criteria that matched the properties contained in the high availability group's name.

This field is read-only.

Use the **Disable** button if you want to prevent all of the members of a high availability group from being activated. One of the few times you might want to use this button is if you are planning to remove or delete the server on which this group is running.

Use the **Enable** button to enable all of the members of a high availability group that were previously disabled. These members can then be activated according to the policy associated with that group.

## High availability group members

Use this page to manage the state of the individual members of a high availability group. This page lists the current members of the selected high availability group.

To view this administrative console page, click **Servers > Core groups > Core group settings > core\_group**. Click on the **Runtime** tab. In the Group name properties field, specify a match criterion for a specific high availability group, or specify an asterisk (\*) to get a list of all the high availability groups that are part of this core group. (A match criterion is one or more name=value pairs that match attributes contained in a high availability group's name.) Then click **Show groups** and select one of the high availability groups listed.

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>    | Specifies the name of a high availability group member.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Node</b>    | Specifies the node on which each high availability group member is running.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Version</b> | Specifies the version of the WebSphere Application Server product on which each node is running.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Status</b>  | Indicates the state of the high availability group members.<br><br>High availability group members are either idle, activated, or disabled. The usual states are idle or activated. One of the few times you might want to disable a member is if it is running on a server that you plan to remove or delete. <ul style="list-style-type: none"> <li>• If a group member is idle, it cannot be assigned any work.</li> <li>• If a group member is activated, it can be assigned work.</li> <li>• If a group member is disabled, it must be enabled before it can be activated.</li> </ul> |

Only use the **Disable** buttons if you want to prevent a group member from being activated.

Use the **Enable** button to enable a group member that was previously disabled.

Use the **Activate** button to activate an idle group member.

Use the **Deactivate** button to make a group member idle.

---

## Creating a policy for a high availability group

Before creating a new policy, you should understand the following requirements for matching a high availability group to a policy:

- A single high availability group cannot match to more than one policy. If the same match criteria is used for multiple policies, the high availability groups with names that match this criteria do not activate when the server is started and an error message, indicating that multiple policies match this group, is logged.
- The same policy can be used for multiple high availability groups.
- A hierarchy is used when matching a high availability group to a policy. If multiple properties are used for the match criteria, the policy that has the most matches with the high availability group name is the policy that is used.

You might want to perform this task if you have a high availability group that requires a policy that is different from the policy governing other high availability groups in the same core group. For example:

- A One of N policy is in effect for all of your high availability groups
- There is a high availability group within the core group that needs to have 5 members active at all times.

You can create a policy with M of N specified as the policy type and 5 specified for the Number of active members, and specify a match criterion that associates this policy with the high availability group that requires this policy.

**Important:**

- Do not change any of the default policies that are shipped with the WebSphere Application Server product. If you need to use different policies, create new policies and specify a match criterion that matches multiple attributes contained in the name of the high availability group. A policy with a greater number of match criterion matches overrides the IBM provided default policies. The IBM provided policies are still available for your use if you encounter a problem with the policies you create.
- Before changing the policy type for a high availability group, make sure you fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus is using a one of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.

To create a new policy:

1. In the administrative console, click **Servers > Core groups > Core group settings**  
*core\_group\_name* > **Policies > New**
2. Select the new policy you want in effect for a specific high availability group. If you need to define a new policy, the policy options are:

Following is a list of valid policy types:

**All active**

The All active policy indicates that the high availability manager keeps all of the application components that are running on all of the servers in the high availability group active at all times.

**M of N**

The M of N policy is similar to the One of N policy. However, it enables you to specify the number (M) of high availability group members that you want to keep active if it is possible to do so. The number of active members must be greater than one and less than or equal to the number of servers in the high availability group. If the number of active servers is set to one, this policy is a match for the One of N policy.

**No operation**

The No operation policy indicates that no high availability group members are activated when the server is started.

**One of N**

The One of N policy provides singleton failover. It keeps a singleton running on one server. If a failure occurs, the high availability manager starts the singleton on another server. One of N is the default policy for WebSphere Partitioning Facility (WPF). If you select this policy, make sure that:

- All external resources are available to all candidate servers.
- A remote database or a Cloudscape database on a network attached storage (NAS) device is available to all servers in the high availability group to enable messaging between these servers.
- The transaction logs are on a NAS device that is available to all servers that are in the high availability group.

**Static**

The Static policy indicates that you must statically define or configure the active members of the high availability group.

Multiple policies can be defined if different high availability groups require different policies. However, only one policy can be associated with a given high availability group. See for more information about these policies.

3. Click **Next**.

4. Specify a name for the policy in the **Name** field. The name must be unique within the scope of the core group. The name should be meaningful to other administrators. For example, if you are setting up a new policy for the service integration bus, you might name the policy My Service Integration Bus Policy.
5. Specify a value for the **Is alive timer** field if the default value is too long or too short a time period. This value determines how frequently the high availability manager checks the health of the high availability group members. The default value is 0 seconds.
  - If you specify -1 the Is alive timer is disabled.
  - If 0 (zero) is specified, the value specified for the Is alive timer at the core group services level is used for high availability groups associated with this policy.
  - If an integer between 1 and 2147483647, inclusive, is specified, this value is used for high availability groups associated with this policy.
6. Select **Quorum** if you want to enable quorum checking. When selected, quorum checking is enabled for a high availability group governed by this policy. Quorum is a mechanism that can be used to protect resources shared across members of the group in the event of a failure.

**CAUTION:**

**Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.**

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.
- **GroupName.WAS\_CLUSTER=cluster\_name** must be specified as one of the name=value pairs contained in the dynamically generated name of a high availability group to which this policy applies. (A component that is using the high availability group function must include the name of its high availability group as part of its component code.)

When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are  $n$  members in the group,  $(n/2) + 1$  servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

7. For M of N and One of N policies, select the Fail back option if, when the most preferred server is available, you want it to take back the workload from the servers that did the failover.
8. For M of N and One of N policies, select the Preferred servers only option if you only want group members activated on servers included in the list of preferred servers for the group. If you select this option, you must set up a list of preferred servers after you click **OK**. A description of how to set up this list is provided in a later optional step.
9. For an M of N policy, specify in the Number of active members field the number of group members that you want to be activated.
10. Click **OK** and then select **Match criteria** .
11. On the next panel, click **New** and then specify one of the name and value pairs that you want to include in the match criterion for this policy. The name and value pair must match one of the name=value attributes contained in the name of a high availability group to which you want to associate this policy.

You, as the WebSphere Application Server administrator, do not have any direct control over the high availability group name. The implementer of the high availability group code decides which properties are used for the group name. You can only control the policy match criterion. “Core group settings” on page 303 describes how to determine the name of a high availability group. that is part of a core group.

You should set the match criterion for a new policy to two or more of the name=value attributes contained in the high availability group’s name to ensure that this policy is used instead of the default policy that IBM provides.

To Specify one of the name and value pairs that you want to include in the match criterion for this policy:

- a. In the Name field, enter the name of one of the name=value attributes contained in the name of the high availability group with which you want to associate this policy. For example, the service integration bus high availability group has the name:

```
IBM_hc=AcceptanceCluster WSAF_SIB_BUS=SSS_Bus
WSAF_SIB_MESSAGING_ENGINE=AcceptanceCluster.000-SSS_Bus
temp_property=WSAF_TEMP type=WSAF_SIB
```

You can specify IBM\_hc, WSAF\_SIB\_BUS, WSAF\_SIB\_MESSAGING\_ENGINE, temp\_property or type in the Name field.

- b. In the Value field, enter the value of the attribute you specified in the Name field. For example, if you specify IBM\_hc in the Name field, you must specify AcceptanceCluster in the Value field for this match criterion to match an attribute included in the name of the service integration bus high availability group.
- c. Optional: Add a description of this match criterion in the Description field. For example, you might specify First attribute to indicate that this name=value pair matches the first attribute contained in the group name.
- d. Click **OK**.
- e. Repeat these steps for each additional attribute you want to include as part of your match criterion. For example, you can specify type for the Name field and WSAF\_SIB for the Value field to include this name and value pair as part of your match criterion for this policy.

Using this example, the following high availability group-to-policy association is established:

| High availability group | High availability group name                                                                                                                            | Policy name                       | Policy match criterion                     |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|--------------------------------------------|
| service integration bus | IBM_hc=AcceptanceCluster<br>WSAF_SIB_BUS=SSS_Bus<br>WSAF_SIB_MESSAGING_ENGINE=AcceptanceCluster.000-SSS_Bus<br>temp_property=WSAF_TEMP<br>type=WSAF_SIB | My Service Integration Bus Policy | IBM_hc=AcceptanceCluster,<br>type=WSAF_SIB |

12. For a Static policy, under **Additional Properties**, select **Static group servers** to select the servers that you want activated. Use **Add** to move core group servers into this list.
13. **Optional:** If you selected the Preferred servers only option, set up a list of preferred servers.
  - a. Under **Additional Properties**, select **Preferred servers**.
  - b. Use **Add** to move core group servers into the list of preferred servers. You can use **Move up** and **Move down** to adjust the order of the servers within the list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.
14. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

The new policy goes into affect after it is saved and synchronized. You do not have to stop and restart the affected application servers. In the future, you can change this policy's settings for the Fail back and Preferred servers only options, and create or update the list of preferred servers associated with this policy without stopping and restarting the affected application servers.

## Core group policies

Use this page to create or update the policy that determines how many members of a high availability group should be active at a given point in time. A high availability group member is active if it is able to accept work.



To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group > Policies*.

Click **New** to define a new policy. After a policy is defined there are several fields that you can no longer change. To change those fields, delete and redefine the policy. Click **Delete** after selecting a policy to delete the selected policy.

All of the policy fields on this page are read-only. To change the values specified in any of these fields, click on the name of the policy you want to change. When the console page **Core group settings > group\_name > Policies > policy\_name** displays, you can edit the policy properties.

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>           | Displays the name of the policy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Description</b>    | Displays the description of the policy.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Policy type</b>    | <p>Specifies the type of policy that is implemented for a high availability group that is associated with this policy.</p> <ul style="list-style-type: none"><li>• <b>All active policy:</b> Under this policy, all of the group members are activated.</li><li>• <b>M of N policy:</b> Under this policy, <i>M</i> group members are activated. The number represented by <i>M</i> is defined as part of the policy details.</li><li>• <b>No operation policy:</b> Under this policy, no group members are activated.</li><li>• <b>One of N policy:</b> Under this policy, only one group member is activated.</li><li>• <b>Static policy:</b> Under this policy, the active members of a group are statically assigned.</li></ul> |
|                       | <b>Restrictions:</b> <ol style="list-style-type: none"><li>1. If you are setting up a policy for a transaction manager, you must select One of N as the policy type.</li><li>2. If you are setting up a policy for a service integration bus you must select One of N or Static as the policy type. The default policy that IBM provides for a service integration bus uses a One of N policy type.</li></ol>                                                                                                                                                                                                                                                                                                                       |
| <b>Match criteria</b> | Specifies one or more name=value pairs that are used to associate this policy with a high availability group. These pairs must match attributes that are contained in the name of a high availability group before this policy is associated with that group.                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Core group policy settings

Use this page to define a policy for a high availability group. Even though a policy is defined at the core group level, it only applies to a high availability group that is contained within the core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or *existing core group > Policies > policy\_name*.

|             |                                                                                              |
|-------------|----------------------------------------------------------------------------------------------|
| <b>Name</b> | Specifies the name of the policy. This name must be unique within the scope of a core group. |
|-------------|----------------------------------------------------------------------------------------------|



|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Policy type</b>    | Specifies the policy type that was selected when this policy was created. This is a read-only field. If you want to change the policy type, you must delete this policy and then create it again specifying a different policy type. If this is an IBM provided policy, do not delete it. Instead create a new policy and specify more of the attributes contained in the high availability group's name as the match criterion for this new policy. The policy with the greatest number of matches to attributes in a group's name is the policy that is associated with that group.                                                                                                                                                                                                                                                                                                                                              |
| <b>Description</b>    | You can use this field to provide meaningful information about this policy. For example, the Clustered TM Policy provided with the product has TM One-Of-N Policy in the description field.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Is alive timer</b> | <p>Specifies the interval of time at which the high availability manager will check the health of the active group members that are governed by this policy. If a group member has failed, the server on which the group member resides is restarted.</p> <ul style="list-style-type: none"> <li>• Valid values are any integer from -1 to 600 seconds, inclusive.</li> <li>• If -1 (negative 1) is specified, this function is disabled.</li> <li>• If 0 (zero) is specified, the high availability manager uses the time interval specified at the application server process level is used.</li> <li>• If a value larger than 0 (zero) is specified, the high availability manager uses the time interval specified here, instead of the one specified at the application server process level, when determining how frequently it should check the health of the high availability group members using this policy.</li> </ul> |

## Quorum

When selected, quorum checking is enabled for a group governed by this policy. Quorum is a mechanism that can be used to protect resources that are shared across members of the group in the event of a failure.

### CAUTION:

**Quorum is an advanced hardware function and should not be enabled unless you thoroughly understand how to properly use this function. If not used properly, this function can cause data corruption.**

The Quorum setting in the policy will only have an effect if the following items are true:

- The group members are also cluster members.
- `GroupName.WAS_CLUSTER=clustername` must be specified as a property in the group name of any high availability group matching this policy.

When enabled, any group using this policy will not achieve quorum until a majority of the members are running. For example, if there are  $n$  members in the group,  $(n/2) + 1$  servers must be online in order to achieve quorum. No group members will be activated until quorum has been achieved.

The quorum mechanism is designed to work in conjunction with a hardware control facility that allows application servers to be shut down if a failure causes the group to be partitioned.

## Fail back

When selected, if a failure occurs, work items assigned to the failing server are moved to the server that is designated as the most preferred server for the group. This field only applies for M of N and One of N policies.

## Preferred servers only

When selected, group members are only activated on servers that are on the list of preferred servers for this group. This field only applies for M of N and One of N policies.

## Number of active members

This field only applies for the M of N policy. Use this field to specify how many of the high availability group members are to be activated.

## New core group policy definition

Use this page to create a new policy for a high availability group.

When you create a new policy, the first page that displays lets you select a policy type. To view the administrative console page where you select a policy type, click **Servers > Core groups > Core group settings > New or select an existing core group > Policies > New**.

Select one of the following policy types:

- All active policy: Under this policy, all of the group members are activated.
- M of N policy: Under this policy,  $M$  group members are activated. The number represented by  $M$  is defined as part of the policy details.
- No operation policy: Under this policy, no group members are activated.
- One of N policy: Under this policy, only one group member is activated.
- Static policy: Under this policy, only specified group members are activated.

See for more information about these policies and restrictions that apply for specific high availability groups.

After selecting a policy, click **Next** to continue.

## Preferred servers

Use this page to define the ordered list of *preferred servers* for the selected policy. The policy gives preference to the servers in this list when activating group members. If there are no preferred servers in the list or none of the servers in the list are active (able to accept work), any available application server will be designated by the coordinator for the high availability group associated with this policy. The coordinator activates and deactivates high availability group members based on the policy that is in effect for that group.

To view this administrative console page, you must be working with a policy that has a policy type of M of N or One of N. If your policy has one of these policy types, click **Servers > Core groups > Core group settings > New >** or *select an existing core group > Policies > New >* or *select an existing policy*. Under **Additional Properties**, select **Preferred Servers**.

Use **Add** and **Remove** to move servers into and out of the list of preferred servers. Use **Move up** and **Move down** to adjust the order within the list of preferred servers. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

Changes to the preferred servers list take affect as soon as they are saved and synchronized. You do not have to stop and restart the affected application servers.

## Preferred coordinator servers

Use this page to define the ordered list of core group servers on which the coordinator servers reside.

To view this administrative console page, click **Servers > Core groups > Core group settings > New >** or *select an existing core group > Preferred coordinator servers*.

Use **Add** and **Remove** to move servers into and out of the list of core group servers on which coordinator servers reside. Use **Move up** and **Move down** to adjust the order within this list. Make sure that the most preferred server is at the top of the list and the least preferred server is at the bottom.

Click **OK** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

## Matching criteria collection

Use this page to view criteria that are defined for a policy.

To view this administrative console page, click **Servers > Core groups > Core group settings > New >** or *select an existing core group > Policies > New* or *select an existing policy > Match Criteria*.

Click **New** to create a new match criteria for the policy. Click the name of a match criteria to change any of that criterion's properties.

|              |                                                   |
|--------------|---------------------------------------------------|
| <b>Name</b>  | Specifies the name of the match criterion.        |
| <b>Value</b> | Specifies the value that is paired with the name. |

**Description**

A description of the match criterion. For example, if you have your own service integration bus policy, the description for the match criterion for that policy might be "My Service Integration Bus Match Criterion."

## Match criteria settings

Use this page to view match criterion defined for a policy.

To view this administrative console page, click **Servers > Core groups > Core group settings > New > or select an existing core group > Policies > New** or *select an existing policy > Match Criteria > criterion name*.

Use the name, value, and description fields to define a match criterion for the policy. The name and value fields should match a name=value attribute included in the name of a high availability group that you want associated with this policy.

**Name**

Specify the name of the match criterion. Create meaningful and consistent criterion names to help define their use.

**Value**

The required value field specifies the value that is paired with the name to determine a match in a policy test.

**Description**

Use the optional description field as a complement to the name field. This field can be particularly useful in environments with multiple system administrators who can describe the test that the criterion is performing.

Click **Apply** to define the criterion. Click **Save** and synchronize changes with nodes after defining new criteria.

## Static group servers collection

Use this page to designate which servers will be made active for the high availability groups associated with a policy that has Static for its policy type.

This option is available under **Additional Properties** only if Static is selected as the policy type. To view this administrative console page, click **Servers > Core groups > Core group settings > existing\_core\_group > Policies > static\_policy\_name > Static group servers**.

Use **Add** and **Remove** to move servers into and out of the list of servers that are designated as static group servers.

Click **Apply** to make your changes effective. Click **Save** to save and synchronize your changes with all managed nodes.

---

## Changing the policy of a high availability group

**Important:**

- Do not change any of the default policies that are shipped with the WebSphere Application Server product. If you need to use different policies, create new policies and specify a match criterion that matches multiple attributes contained in the name of the high availability group. A policy with a greater number of match criterion matches overrides the IBM provided default policies. The IBM provided policies are still available for your use if you encounter a problem with the policies you create.

- Before changing the policy type for a high availability group, make sure you fully understand how the application server processes that are contained in that high availability group are configured and how they will be affected by the policy change. You must also verify that the component that uses that particular high availability group supports the new policy type. For example, if the high availability group for the service integration bus is using a one of N policy, because it only wants one server to be active at any given point in time, and you change the policy associated with that group to All Active, the service integration bus high availability support no longer functions properly and data corruption might occur.
- Adhere to the following requirements for matching a high availability group to a policy:
  - A single high availability group cannot match to more than one policy. If the same match criteria is used for multiple policies, the high availability groups with names that match this criteria do not activate when the server is started and an error message, indicating that multiple policies match this group, is logged.
  - The same policy can be used for multiple high availability groups.
  - A hierarchy is used when matching a high availability group to a policy. If multiple properties are used for the match criteria, the policy that has the most matches with the high availability group name is the policy that is used.

**Example:** If you have two policies, Policy A and Policy B, and

- The match criteria for policy A is `type=WAS_TRANSACTIONS`.
- The match criteria for policy B is both `type=WAS_TRANSACTIONS` and `IBM_hc=TestCluster`.
- When the transaction manager high availability group is created, it is automatically associated with Policy B because it has the closest match to the group name, which is `GN_PS=carbideCell\carbide\ServerA IBM_hc=TestCluster type=WAS_TRANSACTIONS`

To change the policy for a high availability group:

1. Determine if there are any policy restrictions for this high availability group. includes information about any policy restrictions for the IBM provided high availability groups.
2. Create a new policy that has an approved policy type and configuration.
3. Use the **Core Group Runtime** panel in the administrative console to see a list of high availability groups exist for this core group and determine which ones need a new policy. While looking at this panel, you should note the group name properties for all of the groups you are changing.
4. Update the match criteria for each new policy so that it matches the original match criteria and at least one other attribute from the name of the high availability group with which you want to associate the new policy. Two policies can not have exactly the same match criteria because only one policy can be associated with a given high availability group.
5. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

---

## Adding members to a core group

Two or more core groups must be defined for a single cell.

You might need to perform this task after you perform one of the following tasks:

- You create a new core group and want to move some of the members of another core group into this new core group.
- You create a new application server and want to move it to a core group other than the default core group. When a new application server is created, it is automatically added to the default core group.

When adding new members to a core group, remember that:

- All server processes and Java virtual machines (JVMs), including the node agents on all managed nodes, the deployment manager, and application servers within a cell, can only belong to one core group.
  - Core groups cannot overlap each other. If a cluster is defined for the cell, all of the cluster members must belong to the same core group.
1. In the administrative console, click **Servers > Core groups > Core group settings > *core\_group\_name* > Core group servers**.
  2. In the Select column, select the application servers that you want to move.
  3. Click **Move**. The **Core groups > DefaultCoreGroup > Core group servers > Move** administrative console panel is displayed showing the application servers you want to move and the core group to which these application servers currently belong.
  4. Indicate in the To core group field the core group to which you want these application servers moved.
  5. Click **Apply**.
  6. Click **Synchronize changes with Nodes** and then **Save** to save your changes.

The appropriate application servers reside in each core group defined for the cell.

## Core group servers

A core group server is an application server, a deployment manager, or a node agent that is a member of a high availability core group. Use this page to move servers into a different core group. All members of a cluster must be in the same core group. If you select one or more members of a cluster, all of the members of that cluster must be moved.

To view this administrative console page, click **Servers > Core groups > Core group settings > New** or **Select an existing core group for editing > Core group servers**.

|                     |                                                                                                                                                                                                                                                                 |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>         | Specifies the names of the servers in the core group. This field is read-only. Click on this field to the specify custom properties for this server.                                                                                                            |
| <b>Node</b>         | Specifies the node that contains the core group server. This field is read-only.                                                                                                                                                                                |
| <b>Version</b>      | Specifies the product version of the node in the cell. A Version 6 deployment manager node can own a Version 5 managed node. Version 5 managed nodes are easily identified in this field. This field is read-only.                                              |
| <b>Type</b>         | Identifies the server process type, which can be either deployment manager, node agent, or application server. Standalone application servers in the cell, managed nodes, and cluster members all display as application server types. This field is read-only. |
| <b>Cluster Name</b> | Specifies the cluster name if the core group server is part of a cluster. If the core group server does not belong to a cluster, this field is blank. This field is read-only.                                                                                  |

To move one or more servers to another core group, select the check box next to the names of the servers that you want to move and click **Move**.

## Core group server settings

Use this page to create or change custom properties for a server in a core group.

To view this administrative console page, click **Servers > Core groups > Core group settings > Select an existing core group > Core group servers > Select an existing server**.

Extended information about the fields on the panel:

|                          |                                                                                             |
|--------------------------|---------------------------------------------------------------------------------------------|
| <b>Node</b>              | Displays the name of the node that contains the core group server. This field is read-only. |
| <b>Name</b>              | Displays the name of the core group server. This field is read-only.                        |
| <b>Custom Properties</b> | Click on this field to define or edit a custom property for the core group server.          |

## Core group server move settings

Use this page to move one or more servers into a different core group.

Servers, other than the node agent and deployment manager servers, can be moved from one core group to another. However all members of a cluster must be in the same core group. If one or more of the servers you are moving belongs to a cluster, you must move all of the members of that cluster. To view this administrative console page, click **Servers > Core groups > Core group settings > core\_group > Core group servers**. Select the servers to be moved and then click **Move**.

Extended information about the core group fields:

|                              |                                                                                                             |
|------------------------------|-------------------------------------------------------------------------------------------------------------|
| <b>Move selected servers</b> | This field lists the servers that you selected to be moved. It can not be edited.                           |
| <b>From core group</b>       | This field specifies the name of the core group that you are moving the servers from. It can not be edited. |
| <b>To core group</b>         | From the pull-down list, select the core group that these servers will be moved into.                       |

Click **Apply** to make your changes effective. Click **Save** and save and synchronize your changes with all managed nodes.

---

## Routing high availability group work to a different server

How you route work items for a high availability resource to a different application server during runtime depends on how your application server environment is configured.

You might need to perform this task if you have to change the application server that is handling work items for a high availability resource, such as the service integration bus. For example, if a core group includes a cluster of application servers, and a messaging engine is configured for that cluster, any of the servers in that cluster can handle work items for the messaging engine. However, if the high availability group with which the messaging engine is associated is governed by a One of N policy, only one of these application servers can be active at a given point in time.

If you are using the WebSphere Application Server default configuration for a high availability environment, you can change the application server that is handling the work items for a high availability resource. This change can be performed as a run-time operation (the change exists only in memory) or it can be done as a permanent configuration change. To make this change a permanent configuration change, perform one of the following steps:

1. Create a preferred servers list for the policy that governs the high available group. In the administrative console, you can click **Show groups** on the Runtime tab for your core group if you need to determine the high availability group name for the resource that is involved in the change.

For example, to change which server is handling the work items for the service integration bus, create a preferred servers list for the Default SIBus Policy, which is the default policy for the service integration bus high availability group.



- a. To view this administrative console page, click **Servers > Core groups > Core group settings > DefaultCoreGroup > Policies > Default SIBus Policy > Preferred servers**
- b. Add the desired servers to the Preferred servers list.
- c. Click **Save**, select **Synchronize changes with Nodes** and then click **Save** again to save your changes.

As soon as you save your changes, all work items for the service integration bus will be routed to the new preferred server.

2. Create a new static policy for the high availability group to which the resource belongs. Make sure that the match criteria you specify for this new policy includes at least two of the properties specified as part of that high availability group name. To determine the high availability group name for a resource, make sure the application server that is currently handling the work items for the resource is active, and then click **Show groups** on the Runtime tab, in the administrative console, for your core group and enter an asterisk (\*) in the Group name properties field.

For example, to move a messaging engine, create a new policy for the messaging engine high availability group that has Static as the policy type. The match criteria for this policy must include at least two of the properties that are contained in that high availability group name. The two property match will cause the new policy to be used instead of the default policy for this group.

- a. Determine the high availability group name for the messaging engine you want to move. In the administrative console, click **Core groups > Core group settings > DefaultCoreGroup**, and then click on the Runtime tab. Enter an asterisk (\*) in the Group name properties field. A list of the high availability groups that are contained in that core group are displayed in an administrative console panel. An example of a group name that might display on this list follows. The name is split here for printing.:

```
IBM_hc=AcceptanceCluster,WSAF_SIB_BUS=SSS_Bus,WSAF_SIB_MESSAGING_ENGINE=
AcceptanceCluster.000-SSS_Bus,temp_property=WSAF_TEMP,type=jms
```

- b. Create a new static policy. In the administrative console:
  - 1) Click **Core groups > Core group settings > DefaultCoreGroup > Policies > New**
  - 2) Select Static Policy for the policy type and click **Next**.
  - 3) Enter a policy name.
  - 4) Enter a policy description.
  - 5) Default isAlive timer setting should be 0.
  - 6) Do not select Quorum.
  - 7) Click **Apply** to save the new policy.
  - 8) Under Additional Properties, select **Match Criteria** and specify at least two of the properties that exist in the name of the high availability group for the messaging engine. Using the preceding group name, you might specify the following two match criteria:

```
WSAF_SIB_MESSAGING_ENGINE=AcceptanceCluster.000-SSS_Bus
type=jms
```

- 9) Under Additional Properties, select **Static group servers** and specify the name of the application server on which you want the messaging engine to be active.
- 10) Click **OK** and **Save** to save your changes.

New work items for the high availability group are now routed to a different server.

---

## Configuring the core group bridge service

The core group bridge service can be configured for communication between core groups. A core group is a statically defined component of the high availability manager. To configure communication between core groups, use an access point group. An access point group is a collection of core groups that communicate with each other.

Configure two or more core groups that need to communicate with each other. For more information about core groups, see “Core groups” on page 300. To configure core groups, see “Creating a new core group” on page 298.

You must configure the core group bridge service whenever two or more core groups are configured in the same cell. You can also configure the core group bridge to share traffic among core groups that are in different cells. Configure the core group bridge to communicate between cells only when the service is required by another WebSphere Application Server component. By configuring the core group bridge service, the availability status of the servers in each core group is shared among all the configured core groups. For more information, see “Core group communications using the core group bridge service.” You can configure core groups to communicate in the following ways:

- Configure core group communication between core groups that are in the same cell. Configuring communication between any core groups that are in the same cell is required. For more information, see “Configuring communication between core groups that are in the same cell” on page 329.
- Configure core group communication between core groups that are in different cells. You can configure each cell to communicate with one or more other cells. For more information, see “Configuring communication between core groups that are in different cells” on page 334.
- Configure communication between core groups using a proxy peer access point. Sometimes, your core group might not have access to the core group that you want to communicate with. However, if you can access a core group that can communicate with the inaccessible core group, you can create a proxy peer access point. For more information, see “Configuring core group communication using a proxy peer access point” on page 341.

Multiple core groups can communicate with each other.

Continue configuring the high availability environment. See Chapter 13, “Setting up a high availability environment,” on page 293 for more information.

## Core group communications using the core group bridge service

The core group bridge service can be configured to share availability information about internal WebSphere Application Server components between core groups. For example, by configuring the core group bridge service, each core group can be aware of the status of all of the application servers that are configured in all of the core groups. Use access point groups to define the core groups that communicate. Do not use the core group bridge service to share application information among core groups.

A core group is a statically defined component of the high availability manager. Each cell must have at least one core group. WebSphere Application Server creates a default core group called **DefaultCoreGroup** for each cell. For more information about core groups, see “Core groups” on page 300. Two or more core groups can be set up to communicate with each other and share workload management information by defining access point groups. The core groups that communicate can be in the same cell or in different cells.

### Core group bridge overview

To configure communication between core groups, you must configure an *access point group*. An access point group is a collection of the core groups that communicate with each other. Add a *core group access point* to the access point group for each core group that needs to communicate.

A *core group access point* is a collection of server, node, and transport channel chain combinations that communicate for the core group. Each core group has one or more defined core group access points. The **DefaultCoreGroup** has one default core group access point. However, you might consider configuring more than one core group access point for a core group if that particular core group needs to be connected to other core groups that are on different networks.

The node, server, and transport channel chain combinations that are in a core group access point are called *bridge interfaces*. A server that hosts the bridge interfaces is a *core group bridge server*. The transport channel chain defines the set of channels that are used to communicate with other core group bridge servers. Each transport channel chain has a configured port that the core group bridge uses to listen for messages from other core group bridge servers.

Each core group bridge server must have at least one bridge interface for each core group access point. However, to prevent failure of your configuration, configure multiple bridge interfaces for each core group access point. Then, if one server in one of the bridge interfaces fails, the other bridge interface can still receive information and the core group access point remains functional.

If you are configuring communication between core groups that are in the same cell, create one access point group and add a core group access point for each core group that needs to communicate. The following diagram shows an example configuration in a single cell:

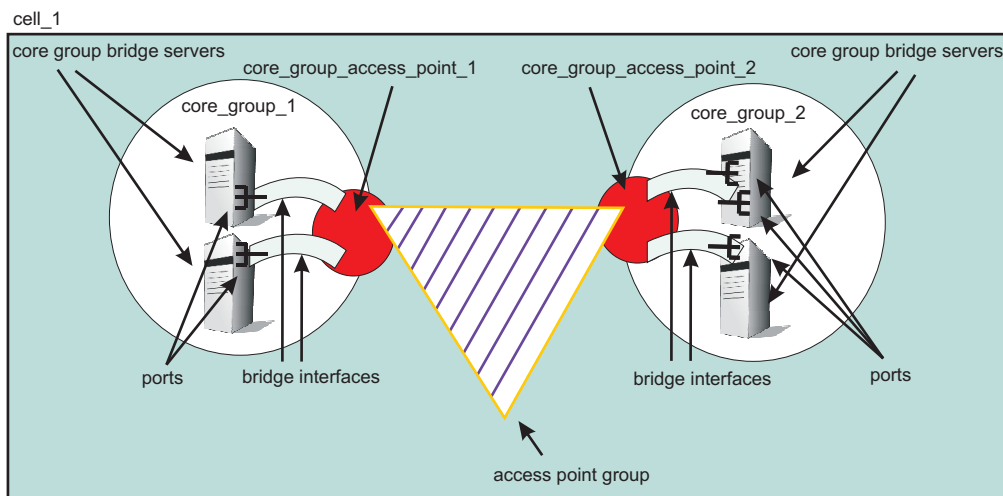


Figure 2. Core group bridge configuration in a single cell

If you are configuring the core group bridge between core groups that are in different cells, you still use an access point group. However, you must create and configure the access point group for each cell. Each cell has an access point group that contains a core group access point for the core group that is in the cell, and a *peer access point* for each peer cell.

A peer access point references a core group access point that is configured in a different cell. Each access point group must have one peer access point for each different cell. Do not configure multiple peer access points that reference the same cell.

Each peer access point has one or more *peer ports* or one *proxy peer access point*.

A peer port corresponds to a bridge interface that is defined in the peer cell. You can define several peer ports for each peer access point.

Define a proxy peer access point if the peer access point cannot be reached directly by using a peer port, but can be reached by using another peer access point. The proxy peer access point specifies a peer access point that can communicate with the peer core group that cannot be reached directly. The proxy peer must have defined peer ports. Specify one proxy peer or one or more peer ports, but not both.

The following diagram shows a core group bridge configuration between two different cells that is using peer access points with peer ports.

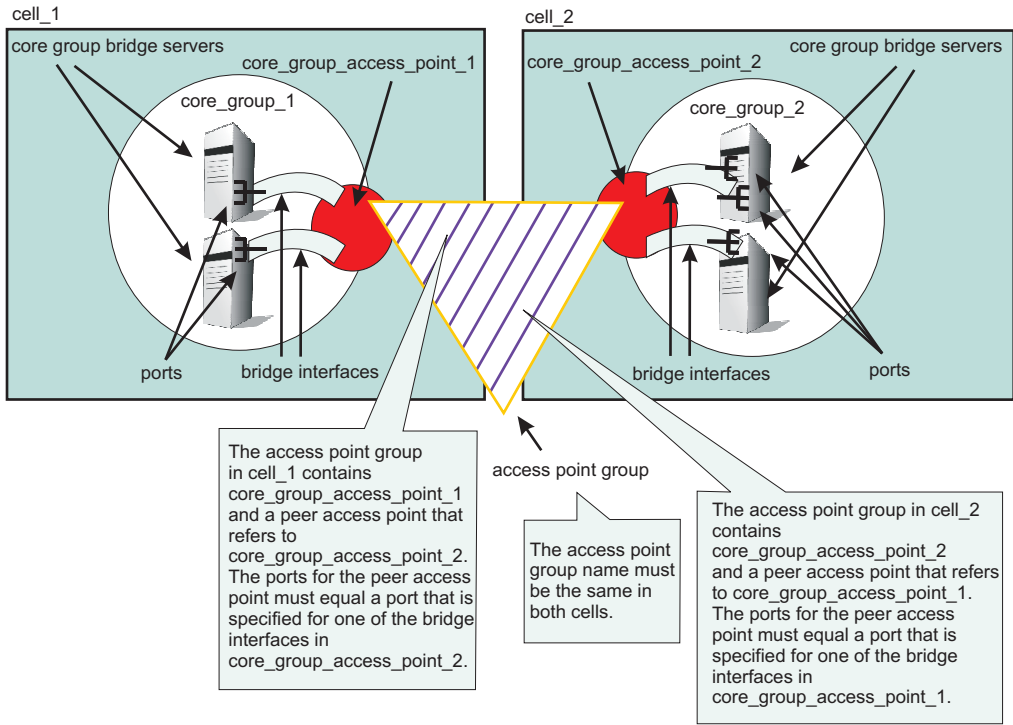


Figure 3. Core group bridge configuration in two different cells

## Configuration scenarios

You can configure core groups to communicate in the following ways:

- “Communication between core groups that are in the same cell”
- “Communication between core groups that are in different cells” on page 327
- “Communication between core groups across different networks” on page 327
- “Communication between core groups using a proxy peer access point” on page 328

### Communication between core groups that are in the same cell

All core groups that are in the same cell must be configured to communicate with each other. To configure core group communication within a cell, create one access point group with one core group access point for each core group. Select one or more servers to be core group bridge servers, and define a bridge interface for each server. The following image shows an example of three core groups that are in the same cell and are connected by one access point group. The sample configuration shows how communication between core groups in the same cell is configured in the administrative console.

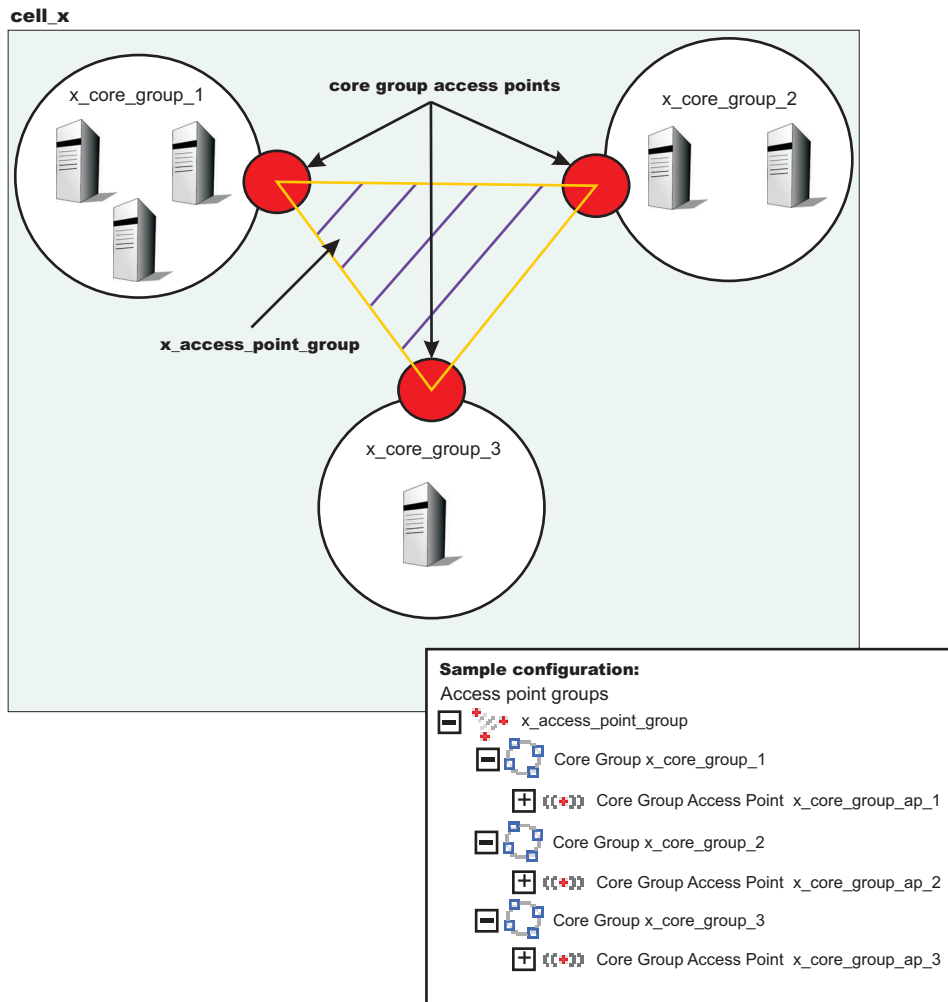


Figure 4. Communication between core groups that are in the same cell

See “Configuring communication between core groups that are in the same cell” on page 329 for more information.

### Communication between core groups that are in different cells

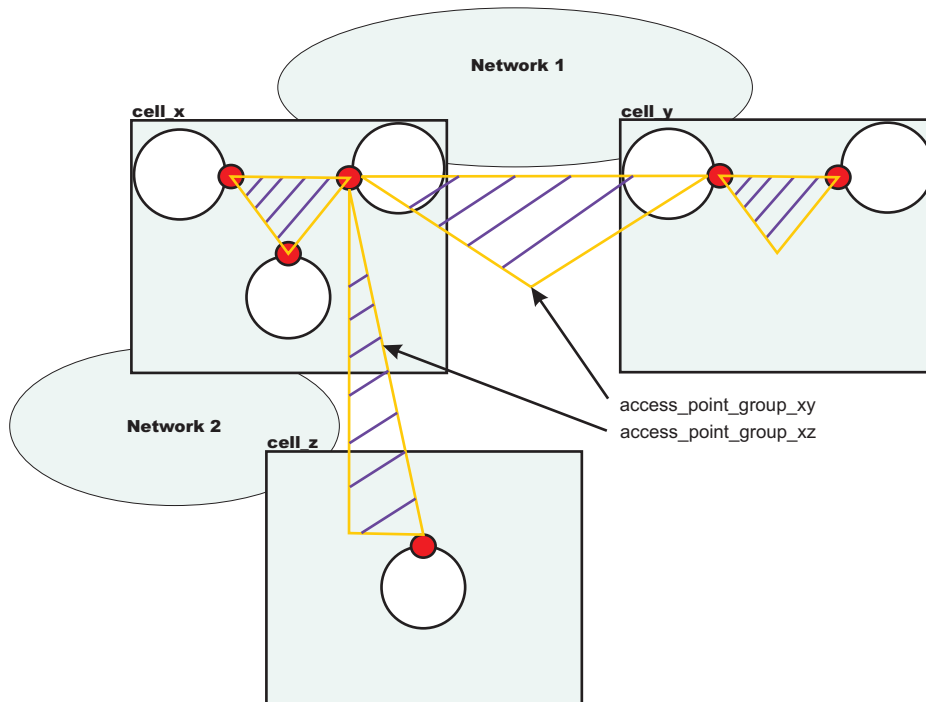
When two core groups in different cells need to communicate, add peer access points to the access point groups. To configure the core group in *cell\_x* to communicate with the core group in *cell\_y*, add a peer access point to the access point group in *cell\_x*. The peer access point is equal to the core group access point for *cell\_y*. Configure an access point group that has the same name for *cell\_y* that contains the core group access point for the core group in *cell\_y* and a peer access point that corresponds to the core group access point for the core group in *cell\_x*. Each peer access point also contains peer ports that identify bridge interfaces in the opposite cells. See “Configuring communication between core groups that are in different cells” on page 334 for more information.

### Communication between core groups across different networks

In this scenario, one core group is configured to communicate with two or more core groups in different cells across two or more networks. For example, a core group in *cell\_x* needs to communicate with core groups in *cell\_y* and *cell\_z*. Create two access point groups in *cell\_x*. The first access point group, *access\_point\_group\_xy*, in *cell\_x* contains a core group access point and a peer access point for the core group in *cell\_y*. The second access point group in *cell\_x*, *access\_point\_group\_xz*, contains a core group access point and a peer access point for the core group in *cell\_z*. *Cell\_y* has an access point group

named *access\_point\_group\_xy*, which has a core group access point and a peer access point for *cell\_x*. *Cell\_z* has an access point group named *access\_point\_group\_xz* which has a core group access point and a peer access point for *cell\_x*.

#### Configuring access point groups across multiple networks



See “Configuring communication between core groups that are in different cells” on page 334 for more information.

#### Communication between core groups using a proxy peer access point

Use a proxy peer when the core groups cannot directly communicate. The two core groups must have access to a single core group that can pass information between the two core groups. To understand what a proxy peer access point does, consider a connecting flight when flying on an airplane. To fly from Pittsburgh to London you first have to fly to New York City, where you change planes and then fly to London. New York City is the *proxy peer access point* for London.

When defining a proxy peer, *x\_core\_group\_2* in *cell\_x* cannot communicate directly with the core group in *cell\_z*. However, both core groups can communicate with the core group in *cell\_y*. To configure communication between *cell\_x* and *cell\_z*, you must configure two access point groups. The core group access point in *cell\_y* is in both of these access point groups, named *access\_point\_group\_xy* and *access\_point\_group\_yz*. The following image shows an overview of a proxy peer configuration.

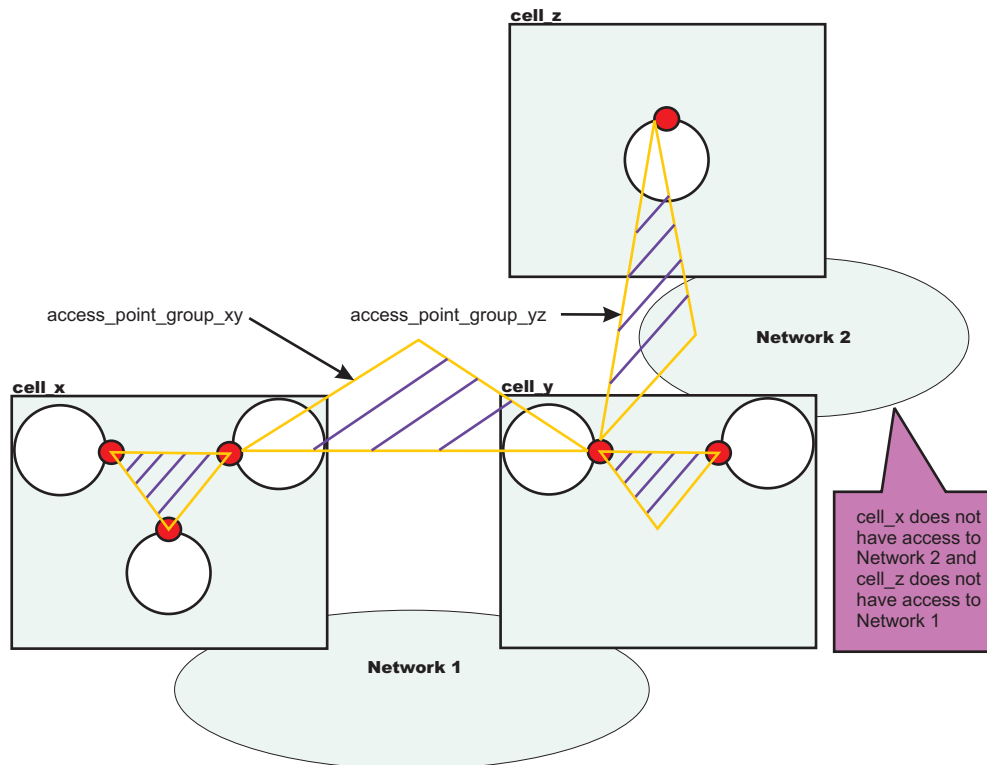


Figure 5. Core group communication using a proxy peer access point

See “Configuring core group communication using a proxy peer access point” on page 341 for more information.

## Configuring communication between core groups that are in the same cell

Use this task to configure communication between core groups that are in the same cell.

Configure two core groups with application servers that are in the same cell. For more information about when and how to configure multiple core groups, see “Creating a new core group” on page 298. Any time you configure multiple core groups that are in the same cell, you must configure communication between the core groups.

Each core group has one or more defined core group access points. A core group access point is a collection of network entry points for the core group. The network entry points that are in a core group access point are called bridge interfaces. An application server that hosts the bridge interfaces is called a core group bridge server.

To configure communication between core groups, define an access point group. An access point group defines the core groups that can communicate with each other. Each access point group has one core group access point for each core group. Select one or more servers to be core group bridges, and define a bridge interface for each server.

See “Core group communications using the core group bridge service” on page 324 for more information about the core group bridge service.

1. Configure an access point group to define the core groups that need to communicate. An access point group contains the core group access points for the core groups that need to communicate. Core group access points define the set of servers that provide access to the core group. To configure



communication between core groups that are in the same cell, you can choose an existing access point group or create a new access point group. To create an access point, complete the following steps:

- a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > New**.
- b. Enter a name for the access point group that is unique within the cell.
- c. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate in the cell. A default core group access point is created whenever a core group is created. The access point group that you create must have a core group access point for each core group in the cell.

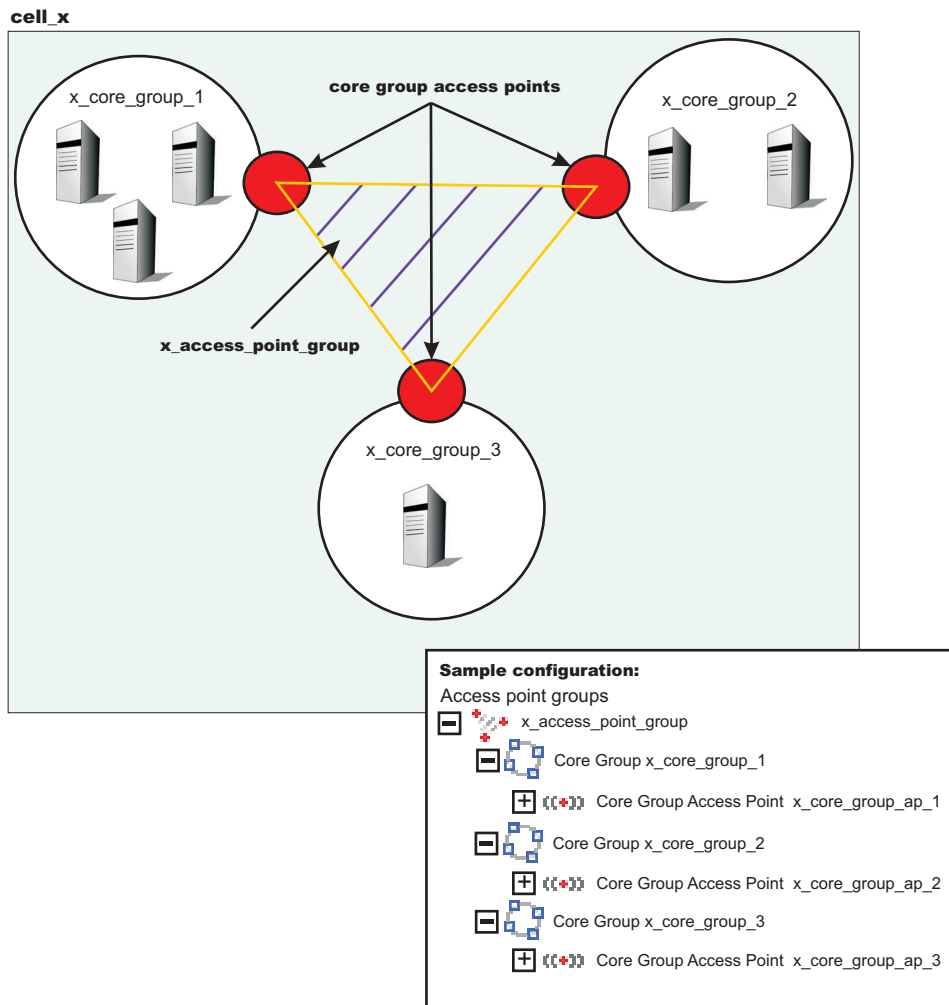
**Restriction:** Do not add any peer access points. Add peer access points only if you are configuring communication with a core group in a different cell. If you need to communicate with core groups that are outside of the cell, you must create another access point group that has one core group access point and one or more peer access points. See “Configuring communication between core groups that are in different cells” on page 334 for more information.

If you use an existing access point group, choose an access point group that does not have peer access points. To configure an existing access point group, perform the following steps:

- a. In the administrative console, click **Servers > Core groups > Core group bridge settings**. Your current configuration with any existing access point groups is displayed.
  - b. Verify that the access point group does not have any peer access points. Peer access point groups are used for communication between core groups in different cells. Click the access point group you want to configure and ensure that no peer access points are listed.
  - c. Click **Access point groups > access\_point\_group\_name > Core group access points**.
  - d. Add core group access points to your access point group. Choose any available core group access points for the core groups that need to communicate. The access point group you create should have a core group access point for each core group in the cell.
2. Create bridge interfaces for each core group access point. The bridge interfaces that you add provide access to the core group. Create at least one bridge interface for each core group access point. To back up your configuration, you should configure two or more bridge interfaces for each core group access point. If a core group has multiple core group access points, each core group access point must contain the same number of bridge interfaces for the same set of servers. To configure bridge interfaces, perform the following steps:
- a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Core group access points**.
  - b. Click a core group access point in the access point group. Click **Show Detail**.
  - c. To create a new bridge interface, click **Bridge interfaces > New**.
  - d. Select a node, server, and transport channel chain combination for the bridge interface. Click **OK**. All the core group access points in the access point group must have transport channel chains with the same port name. The transport channel chain can be the DCS or DCS-secure channel chains that are created for the DCS\_UNICAST\_ADDRESS transport chain.
  - e. Consider creating at least two bridge interfaces for each access point. If one bridge interface fails, the other can still be active.
  - f. Repeat these steps to create bridge interfaces for each core group access point in your access point group.

The core groups that are in the same cell and configured in an access point group can communicate.

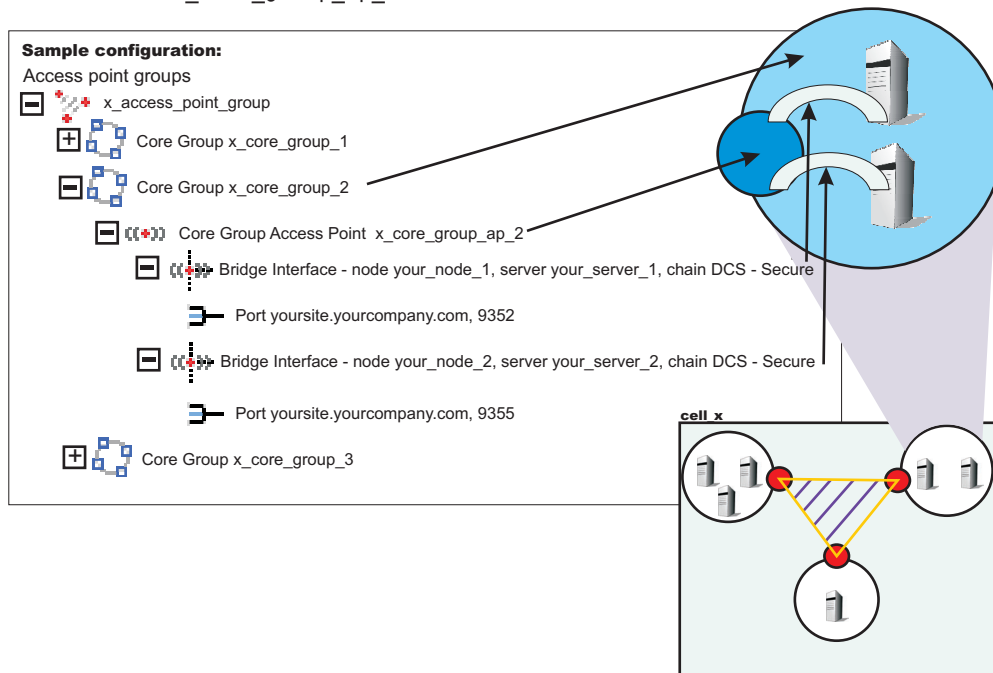
In *cell\_x*, there are three core groups: *x\_core\_group\_1*, *x\_core\_group\_2*, and *x\_core\_group\_3*. Each core group already has a core group access point. The following image shows an access point group between the core groups in *cell\_x* and an example of the configuration in the administrative console.



Perform the following steps to configure communication between the three core groups in *cell\_x*:

1. Create an access point group named *x\_access\_point\_group*. Add a core group access point to the access point group for each core group that is in the cell. In this example, add *x\_core\_group\_ap\_1*, *x\_core\_group\_ap\_2*, and *x\_core\_group\_ap\_3* to *x\_access\_point\_group*.

2. Create bridge interfaces for each core group access point. The following diagram shows the bridge interfaces for `x_core_group_ap_2`:



Create two or more bridge interfaces for each core group access point.

By creating an access point group and adding all core groups in the cell to the access point group, you enabled communication between all the core groups that are in `cell_x`.

You can configure this cell to communicate with core groups in other cells. See “Configuring communication between core groups that are in different cells” on page 334 and “Configuring core group communication using a proxy peer access point” on page 341 for more information.

## Core group access point settings

Use this page to configure your core group access points. The core group access point defines the set of core group bridge servers that provide access to the core group. A core group bridge server is an application server that is configured to run the core group bridge service. Define unique core group access points for each different network over which you want the core group in this cell to connect to other core groups that are defined in another cells. The core group access point has a collection of bridge interfaces. Each server that is used as a bridge must have a unique bridge interface for every core group access point in the core group.

For example if you define access points `access_point_a` and `access_point_b` and servers `server_x` and `server_y` are configured as core group bridges in a core group, `server_x` must have unique bridge interfaces for both `access_point_a` and `access_point_b`. The `server_y` server must also have unique bridge interfaces for both `access_point_a` and `access_point_b`.

When you create a core group, a core group access point is automatically created. Do not delete the last access point in a core group.

The core group access point that is automatically defined belongs to a default access point group. You can use the default core group access point and access point group to configure communication between core groups. You must create and configure bridge interfaces for the default core group access point. See “Bridge interface settings” on page 333 for more information about bridge interfaces.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *core\_group\_access\_point\_name* > Show detail.**

**Name:**

Specifies the name for the core group access point.

**Core group:**

Specifies the core group that is associated with this core group access point.

### Core group access point collection

Use this page to configure your set of core group access points. Core group access points define the set of servers that provide access to the core group. At least one core group access point must be defined for each core group in the local cell.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points.**

**Available core group access points:**

A list of core group access points that are available to add to the access point group.

**Core group access points in access point group:**

A list of core group access points that are in the specified *access point group*.

### Bridge interface collection

Each core group access point has a collection of bridge interfaces. This collection defines the interfaces on the set of servers that provide access to the core group. All the servers in this collection have the core group bridge service enabled. The core group bridge service provides communication between core groups. A bridge interface defines the node, the server, and the chain for the core group access point. The chain defines the transport channels that are used by the server for receiving information.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *access\_point\_name* > Show detail > Bridge interfaces.**

**Server:**

Specifies the node and the server combinations that are bridge interfaces for the core group access point.

**Chain:**

Specifies the transport channel chain that is used for transport by the bridge interface. For all the core group access points in an access point group, the transport channel chains must resolve to the same host. To ensure that the transport channel chains resolve to the same host, use the same chain for all of the core group access points in an access point group.

### Bridge interface settings

A bridge interface is a particular node and server that runs the core group bridge service. The core group bridge service is the service that provides communication between core groups. A bridge interface is defined by a unique combination of a node, server, and transport chain. You cannot configure a cluster of servers to run the same bridge interface. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *access\_point\_access\_point\_name* > Show detail > Bridge interfaces > *server\_node***.

**Bridge interfaces:** Displays a list of server, node, and chain combinations that are available to become bridge interfaces for your core group access point.

**Node:** Displays the node on which the core group bridge service is running.

**Server:**

Displays the application server on which the core group bridge service is running. The server you select can have other responsibilities as well as being a core group bridge server. The server does not have to be running as a standalone node.

**Chain:**

Displays the transport channel chain that is used by the core group bridge service for this core group access point. Transport channel chains contain hosts and ports. The host and port names must be identical for all chains in all core group access points in an access point group. The transport channel chain can be the DCS or DCS-secure channel chains that are created for the DCS\_UNICAST\_ADDRESS transport chain for each application server.

## Bridge interface creation

A bridge interface specifies a particular node and server that runs the core group bridge service. A bridge interface is defined by a unique combination of a node, a server, and a transport chain. A transport chain represents a network protocol stack that is operating within an application server.

To access this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points > *access\_point\_name* > Show detail > Bridge interfaces > New**.

**Available bridge interfaces:**

Specifies the node, server and transport channel chain combinations that are available to become bridge interfaces for this core group access point. Only bridge interfaces with transport chains that contain Transmission Control Protocol (TCP) inbound channels using the same port name as existing bridge interfaces in this core group access point are displayed. The bridge interfaces that are already used by any core group access point are not displayed.

## Configuring communication between core groups that are in different cells

Use this task to configure core group communication between core groups that are in different cells.

A core group is a statically defined component of the high availability manager. For more information about core groups, see “Core groups” on page 300. For this task, configure core groups that are in different cells. For more information about when and how to create multiple core groups, see “Creating a new core group” on page 298. Before you configure a core group to communicate outside a cell, enable core group communication between the core groups that are within each cell. For more information, see “Configuring communication between core groups that are in the same cell” on page 329.

Configure the core group bridge service between cells to share the availability status of the servers in each core group among all the configured core groups. Enable the core group bridge service to communicate between cells only when it is required by another WebSphere Application Server component.

The core group bridge service is a requirement when configuring an Enterprise JavaBeans (EJB) backup cluster. Configure a backup EJB cluster in a different cell to continue servicing requests when the primary cluster fails. See “Creating backup clusters” on page 271 for more information.

When you configure communication between core groups that are in different cells, you must configure an access point group that has the same name in each cell. Each access point group has exactly one core group access point and one or more peer access points. The peer access point corresponds to the core group access point that is in the other cell. Configure one access point group between the cells. Do not create multiple access point groups that allow communication between the same two cells.

Access point groups and access points are abstractions. Therefore, they are not considered a point of failure. An access point fails only if all of servers fail that are specified in the bridge interfaces for that access point. Create one or more bridge interfaces for each core group access point to prevent failure of access point groups. See “Configuring communication between core groups that are in the same cell” on page 329 for more details about bridge interfaces.

Create separate access point groups for each unique network over which the core groups communicate.

1. Configure an access point group that has peer access points for a core group in a different cell. You can select an access point group or create a new access point group. To create a new access point group perform the following steps:
  - a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > New**.
  - b. Type a **Name** for the access point group that is unique within both of the cells.
  - c. Add one core group access point for the core group in the local cell. When configuring an access point group that communicates outside of the cell, always configure exactly one core group access point and one or more peer access points.
  - d. Add an existing peer access point for the core group in the remote cell.
  - e. Confirm the changes to save your new access point group.

To add peer access points to an existing access point group, perform the following steps:

- a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Core group access points**. Check that this access point group contains only one core group access point. You cannot configure an access point group that communicates both inside and outside of the cell. Any access point group for core group communication outside of the cell must contain one core group access point and one or more peer access points.
- b. Add existing peer access points or create a new peer access point. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Peer access points**. To add an existing peer access point, move the peer access point to your access point group. To create a new peer access point, click **New**. The name that you enter for the peer access point must be unique in the cell. To create new peer access points, you must have the following configuration information for the core group in the remote cell:
  - The name of the cell for the core group
  - The name of the core group
  - The core group access point for that core group
  - The host and port information for the core group

When you save the new peer access point, it is automatically added to your access point group.

2. Configure peer ports for each peer access point. A peer port identifies the host name and port of an application server that is defined as a bridge interface in the other cell, so find the host name and port information before completing this step.

- a. In the administrative console, click **Servers > Core groups > Core group communications > Access point groups > *access\_point\_group\_name* > Peer access points**. Select a peer access point that is in your access point group and click **Show Detail**.
- b. Enable this peer access point to use peer ports. Select **Use peer ports**.
- c. Configure the peer ports. Click **Peer ports**. You can select and remove existing peer ports, or create new peer ports.

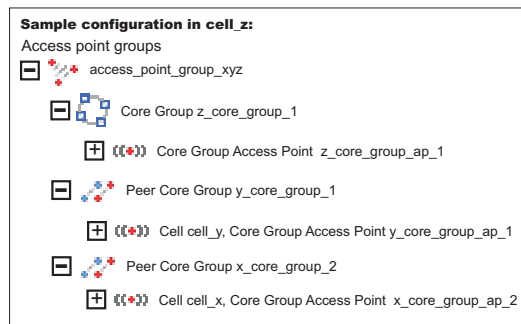
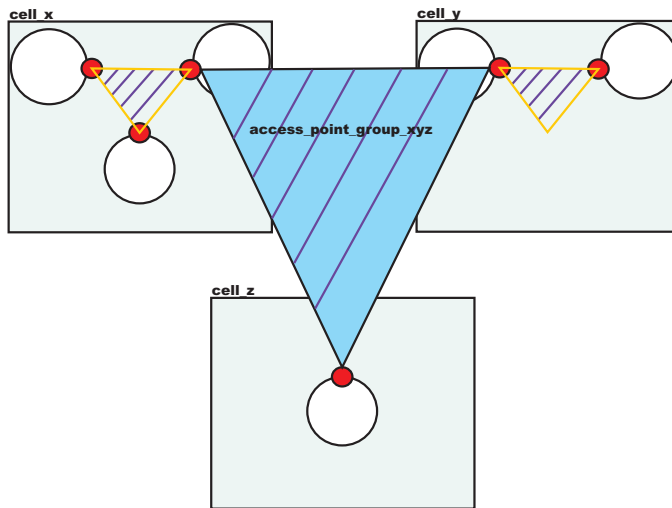
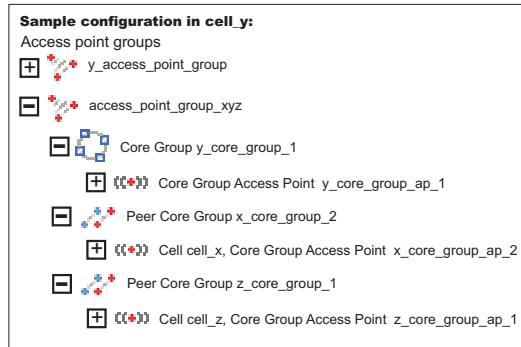
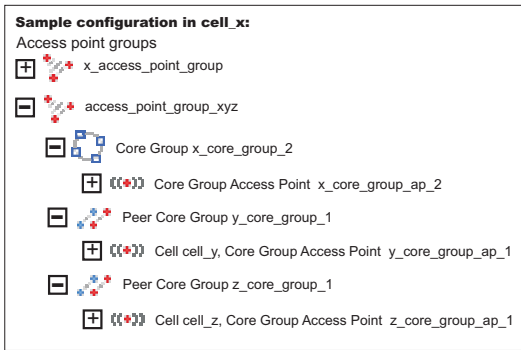
Configure only one peer access point for each cell that you need to communicate with. Do not configure multiple peer access points in one cell that correspond to the same remote cell.

3. Repeat these steps for the core group in the other cell. The access point group that you create must have the same name as the access point group that you defined in the first cell. The peer access points that you create in this access point group correspond to the core group access point in the first access point group.

When you configure core groups in different cells to communicate with each other, all the core groups in each cell receive information from the core groups in the other cells.

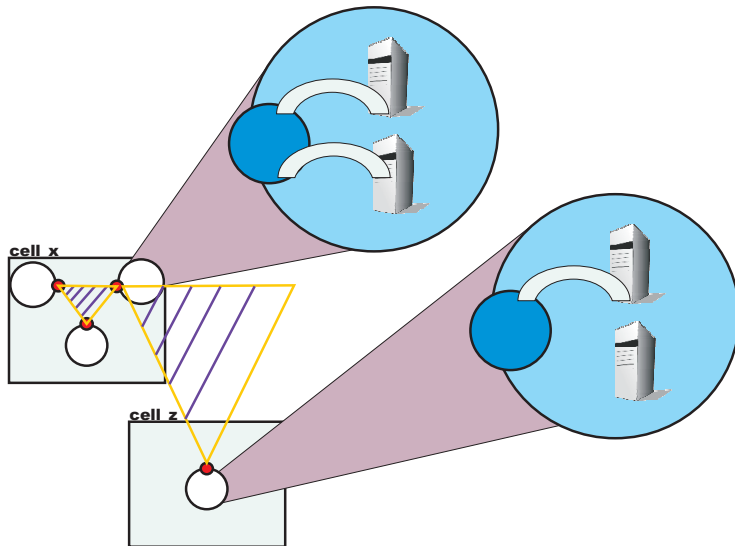
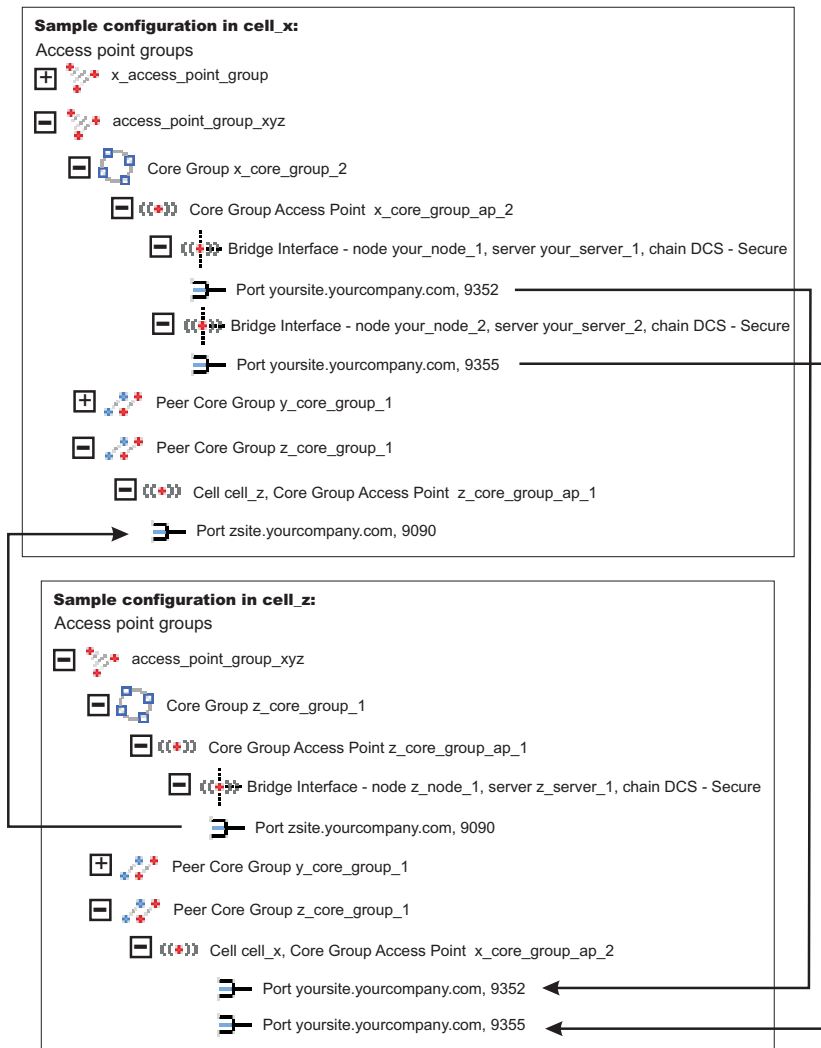
Following is an example of a configuration between three core groups that are in three different cells. Each cell has one access point group for communication between core groups in the cell. Each cell also has a defined access point group, *access\_point\_group\_xyz*, which contains one core group access point group for the core group that is in the cell, and one core group access point for each of the core groups in the other two cells.





The following sample shows the relationship between bridge interfaces and peer ports for the communication between *cell\_x* and *cell\_z*. In *cell\_x*, there are two bridge interfaces defined. In *cell\_z*,

there is a peer access point for *x\_core\_group\_ap\_2* that has peer ports defined that correspond to the bridge interface information that is defined in *cell\_x*.



As a result, *core\_group\_x* , *core\_group\_y* and *core\_group\_z* can communicate with each other.

Continue configuring core group communication and setting up the high availability environment. See “Configuring the core group bridge service” on page 323 and Chapter 13, “Setting up a high availability environment,” on page 293 for more information. To configure a backup EJB cluster, see “Creating backup clusters” on page 271.

### Peer access point settings

Each peer access point is used to communicate with core groups in other cells. A peer access point corresponds to a core group access point in the peer cell. The peer access point communication settings are specified by using one or more peer end points or a proxy peer.

A peer access point must contain either peer ports or a proxy peer access point, but not both. When the peer access point is directly accessible within its access point group, specify peer ports. When the peer access point can be reached only indirectly, use a proxy peer access point. A proxy peer access point is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer access point specifies a peer access point that can communicate with the appropriate destination core group. The specified proxy peer access point must be a peer access point that has defined ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Peer access points > peer\_access\_point\_name > Show detail**.

**Name:**

Specifies the name of the peer access point. The name must be unique within the local cell.

**Cell:**

Specifies the cell in which the peer access point resides.

**Core group:**

Specifies the core group in which the peer access point resides.

**Core group access point:**

Specifies the name of the core group access point that is in the peer cell.

|         |                             |
|---------|-----------------------------|
| default | defaultCoreGroupAccessPoint |
|---------|-----------------------------|

**Use peer ports:**

Specifies that you are using peer ports instead of a proxy peer access point. Use peer ports when the peer access point is directly accessible within its access point group. Click **Peer ports** to specify the peer ports for the peer access point.

**Use proxy peer access point:**

Specifies that you are using a proxy peer access point instead of peer ports. A proxy peer is defined when the peer access point can be reached only indirectly through another peer access point. A proxy peer is used to identify the communication settings for the peer access point that cannot be accessed directly. The proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined peer ports.

**Proxy peer access point:**

Specifies the specific peer access point that is used to access a core group.

### **Peer access point collection**

Peer access points are used to communicate with core groups that are in other cells. A peer access point collection is the set of peer access points that are used to communicate with the core group access point in this access point group. Specify a single peer access point for each remote cell. This collection cannot contain two peer access points for the same cell.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Peer access points**.

#### ***Available peer access points:***

A list of peer access points that are available to join the access point group.

***Peer access points in *access\_point\_group*:*** A list of peer access points that are in the selected access point group.

### **Peer port settings**

Use this page to define a peer port. A peer port identifies the host name and port of an application server that is a bridge interface in another cell. This application server is using the core group bridge service to communicate with other core groups. Each peer access point can have one or more peer ports. Each port identifies a bridge interface of a core group bridge service in the peer cell.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Peer access points > *peer\_access\_point\_name* > Show detail > Peer ports > *peer\_port\_name***.

#### ***Host:***

Specifies the host name on which the core group bridge in the remote cell is listening.

#### ***Port:***

Specifies the port number that is associated with the host on which the core group bridge in the remote cell is listening.

### **Peer port collection**

Use this page to define the peer ports for the peer access point. Each peer port identifies a bridge interface of a core group bridge service in the peer cell. Each peer access point that does not have a proxy peer must have one or more peer ports.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name* > Peer access points > *peer\_access\_point\_name* > Show detail > Peer ports**.

#### ***Host:***

Specifies the host name that is used by the bridge interface in the remote cell.

#### ***Port:***

Specifies the port that is used by the bridge interface in the remote cell.

## Configuring core group communication using a proxy peer access point

Use this task to configure communication between two core groups when they cannot communicate with each other directly through peer ports.

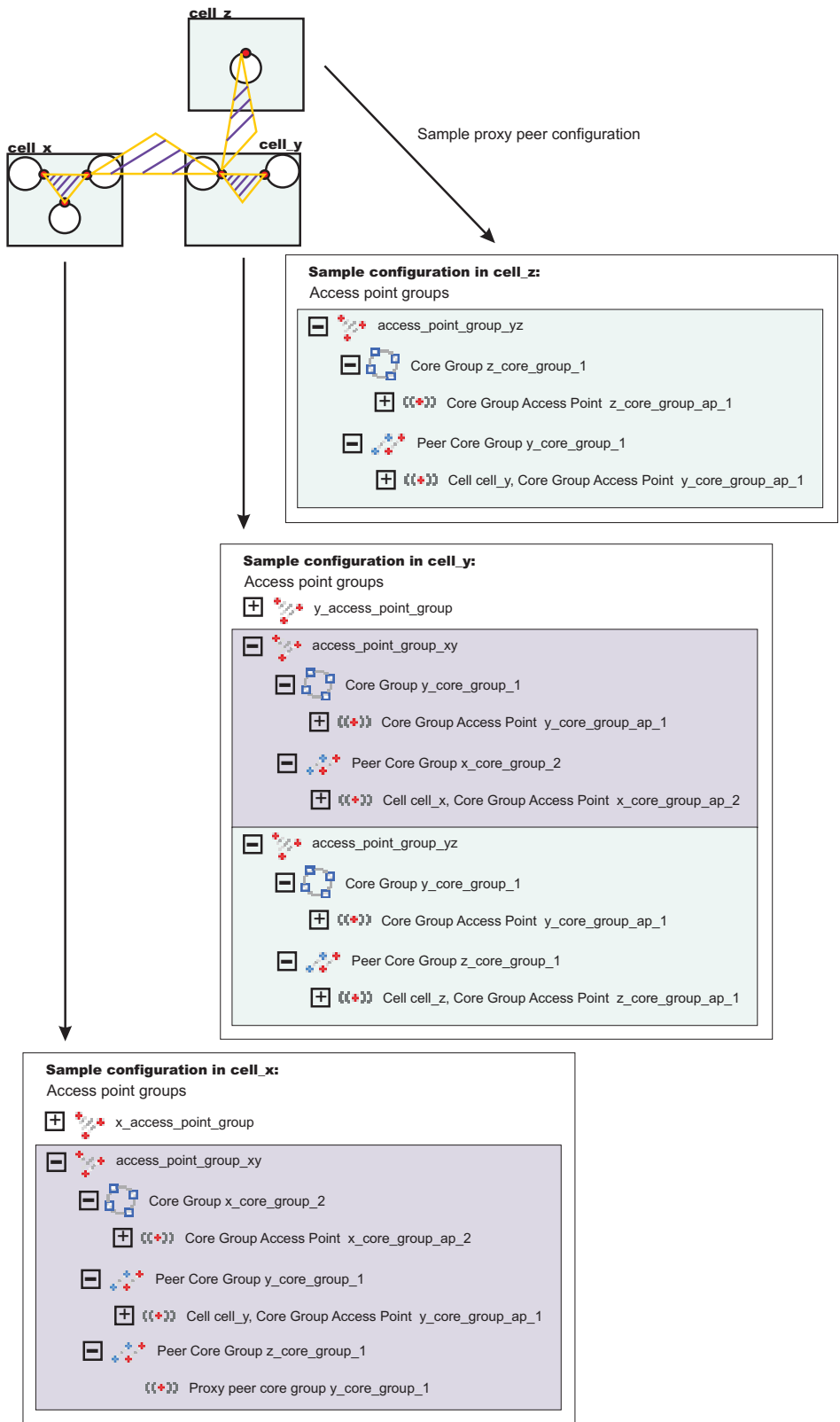
Configure a proxy peer access point to communicate with a core group if you cannot use peer ports. Use a core group that has configured a peer access point with peer ports to the core group that you want to communicate with. Before completing this task, make sure that you have access to a core group that can communicate with the core group that you cannot communicate with directly. To configure communication between core groups that are in different cells, see “Configuring communication between core groups that are in different cells” on page 334. If there are multiple core groups in each of the cells, they must be configured to communicate with each other. See “Configuring communication between core groups that are in the same cell” on page 329 for more information.

Use a proxy peer to communicate with a core group that you cannot access directly. This task describes how to configure a proxy peer with three core groups that are each in different cells. *Core\_group\_x* and *core\_group\_y* can communicate directly with each other through peer ports. *Core\_group\_y* and *core\_group\_z* can also communicate with each other through peer ports. However, *core\_group\_x* cannot communicate with *core\_group\_z*. To establish that communication, *core\_group\_x* has a peer access point that is a proxy peer. The proxy peer is the peer access point to *core\_group\_y*. For more information, see “Core group communications using the core group bridge service” on page 324.

1. Configure *core\_group\_x* and *core\_group\_y* to communicate with each other by creating an access point group. For more information, see “Configuring communication between core groups that are in different cells” on page 334.
2. Configure *core\_group\_y* and *core\_group\_z* to communicate with each other by creating another access point group. For more information, see “Configuring communication between core groups that are in different cells” on page 334. When you do this step, create a second access point group in cell 2. *Core\_group\_2* communicates with both *cell\_x* and *cell\_z* over two different networks.
3. Configure a peer access point that has a proxy peer. Create a new peer access point in the access point group that you created between *core\_group\_x* and *core\_group\_y*.
  - a. In the administrative console, click **Servers > Core groups > Core group bridge settings > Access point groups > access\_point\_group\_name > Peer access points**. Create a new peer access point, or select an existing access point.
  - b. Enter a unique name for the peer access point. For the other cell, core group, and core group access point values, use the properties of *core\_group\_z*.
  - c. Click **Use a proxy peer access point**. Select the proxy peer access point that is the peer access point that you created in *cell\_x* that refers to the core group access point in *cell\_y*.

*Core\_group\_x* can communicate with *core\_group\_z* by using a proxy peer.

The following example shows the configurations in each cell when you configure communication between *cell\_x* and *cell\_z* using a proxy peer access point:



By completing this task, you enabled one-way communication between *core\_group\_x* and *core\_group\_z*. If you want to configure communication both ways, you must repeat these steps, configuring a peer access point in *core\_group\_z* that contains a proxy peer.

## Core group communication

The core group bridge is the service that enables communication between core groups. A core group is a statically defined component of the high availability manager. Use this page to view the structure of your access point groups. Access point groups link core groups that are in the same cell or in different cells and allow the core groups to communicate with each other.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings**.

Each access point group is a collection of core group access points. If you are configuring communication between core groups that are in the same cell, configure an access point group with a core group access point for each core group in the cell. If you are configuring communication between core groups in different cells, configure an access point group that has one core group access point for the local cell and a peer access point for each other cell.

Each core group access point has one or more bridge interfaces. Each peer access point has a proxy peer or one or more peer ports. A bridge interface is a server that is configured to communicate with other core groups by using a particular transport channel chain. Click **Access point groups** to configure the settings for each access point group that is configured.

- **Access point group** - An access point group defines the core groups that communicate with each other. Each access point group consists of a collection of core groups.
  - **Core group** - Specifies a core group that is in this access point group. Core groups are referenced by core group access points.
    - **Core group access point** - The core group access point defines the set of servers that provide access to the core group. Bridge interfaces define the servers that are in the core group access point.
      - **Bridge interface** - A bridge interface defines a node, server, and chain for the core group access point. The chain defines the transport channels that the server uses for receiving information. Typically, all the bridge interfaces in a core group access point use the same chain.
  - **Peer core group** - Specifies a core group in a different cell. Define peer access points to communicate with peer core groups.
    - **Peer access point** - Each peer access point is used to communicate a core group in a different cell. Each peer access point corresponds to a core group access point that is in the peer cell. Use one or more peer ports or one proxy peer to define the communication settings.
      - **Peer port** - Each peer port identifies a bridge interface of a core group bridge in the peer cell.
      - **Proxy peer** - A proxy peer is used to identify the communication settings for a peer access point that cannot be accessed directly through peer ports. A proxy peer specifies a peer access point that can communicate with the destination core group. The specified proxy peer must be a peer access point that has defined ports.

## Access point group collection

Use this page to view the sets of access point groups. Access point groups define the set of core groups that communicate with each other. Access point groups that connect multiple cells must have one core group access point and a single peer access point for each remote cell. Access point groups that provide communications between core groups in the same cell must contain only core group access points.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups**.

### Name

Specifies the name of the access point group. The access point group name must be unique within the cell.



## Access point group settings

Use this page to modify the core group access points and the peer access points that belong to this access point group. An access point group defines the set of core groups that communicate with each other. Group the access points to support communication. Access points can be either peer access points or core group access points. Define core group access points so that core groups in the same cell can communicate. Define peer access points so that core groups in different cells can communicate.

To view this administrative console page, click **Servers > Core groups > Core group bridge settings > Access point groups > *access\_point\_group\_name***.

From this page, you can edit the core group access points or the peer access points that belong to the access point group.

### Name

Specifies the name of the access point group. The access point name must be unique within a cell.

---

## 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 Licensing  
IBM Corporation  
500 Columbus Avenue  
Thornwood, New York 10594 USA



---

## **Trademarks and service marks**

For trademark attribution, visit the IBM Terms of Use Web site (<http://www.ibm.com/legal/us/>).