uCosminexus Application Server

Application Setup Guide

3021-3-J13-10(E)

Notices

■ Relevant program products

See the *Release Notes*.

■ Export restrictions

If you export this product, please check all restrictions (for example, Japan's Foreign Exchange and Foreign Trade Law, and USA export control laws and regulations), and carry out all required procedures.

If you require more information or clarification, please contact your Hitachi sales representative.

■ Trademarks

HITACHI, Cosminexus, HiRDB, OpenTP1, TPBroker, XDM are either trademarks or registered trademarks of Hitachi, Ltd. in Japan and other countries.

Microsoft, SQL Server are trademarks of the Microsoft group of companies.

Microsoft, Windows are trademarks of the Microsoft group of companies.

Microsoft, Windows Server are trademarks of the Microsoft group of companies.

Microsoft is a trademark of the Microsoft group of companies.

Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

UNIX is a trademark of The Open Group.

Other company and product names mentioned in this document may be the trademarks of their respective owners.

Eclipse is an open development platform for tools integration provided by Eclipse Foundation, Inc., an open source community for development tool providers.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

■ Issued

Aug. 2022: 3021-3-J13-10(E)

■ Copyright

All Rights Reserved. Copyright (C) 2022, Hitachi, Ltd.

Preface

For details on the prerequisites before reading this manual, see the *Release Notes*.

■ Non-supported functionality

Some functionality described in this manual is not supported. Non-supported functionality includes:

- Audit log functionality
- Compatibility functionality
- Cosminexus Component Transaction Monitor
- Cosminexus Reliable Messaging
- Cosminexus TPBroker and VisiBroker
- Cosminexus Web Service - Security
- Cosminexus XML Security - Core functionality
- JP1 linkage functionality
- Management Server management portal
- Remote installation functionality for the UNIX edition
- SOAP applications complying with specifications other than JAX-WS 2.1
- uCosminexus OpenTP1 linkage functionality
- Virtualized system functionality
- XML Processor high-speed parse support functionality

■ Non-supported compatibility functionality

"Compatibility functionality" in the above list refers to the following functionality:

- Basic mode
- Check of JSP source compliance (cjjsp2java) with JSP1.1 and JSP1.2 specifications
- Database connection using Cosminexus DABroker Library
- EJB client application log subdirectory exclusive mode
- J2EE application test functionality
- Memory session failover functionality
- Servlet engine mode
- Simple Web server functionality
- Switching multiple existing execution environments
- Using EJB 2.1 and Servlet 2.4 annotation

Contents

1 Overview of Application Setup

This chapter describes the purpose and positioning of setup and the operations that are possible as a part of setup.

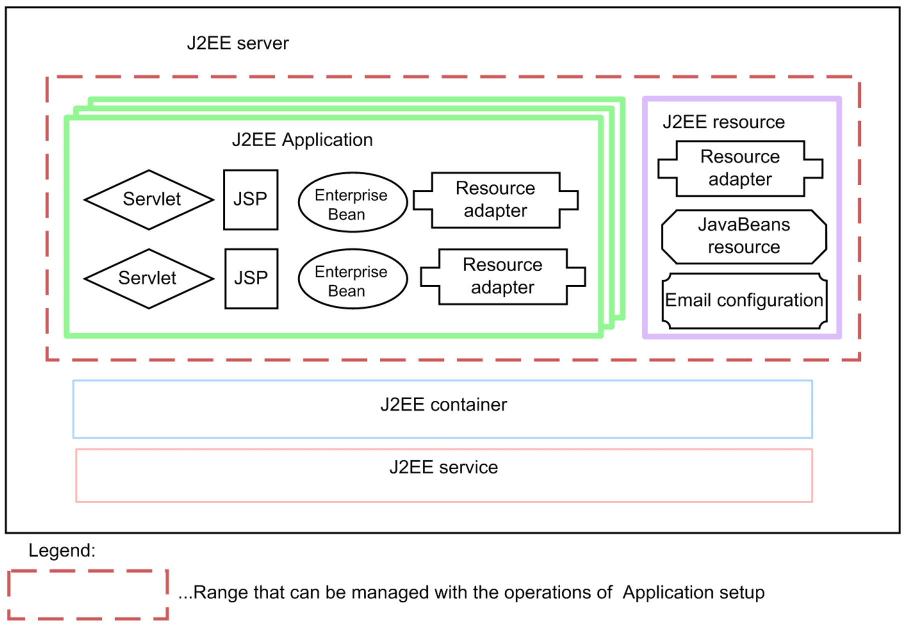
1.1 Purpose of setup

Application setup refers to the setup and customization that is required for proper operation of applications and resources on a J2EE server. Specify the settings in the following cases:

- When setting up resources or J2EE applications to build and modify the operating environment of Cosminexus systems
- When setting up resources or J2EE applications to develop J2EE applications

The following figure shows the configuration of a J2EE server and the scope of the architecture managed by setup.

Figure 1–1: Architectural scope managed by setup



Use the server management commands to execute the application setup. For details on the server management commands, see *2.1 Functionality of server management commands*.

1.2 Managing J2EE resources

This section gives an overview of managing J2EE resources, when performing the application setup.

1.2.1 Types of J2EE resources to be managed

The J2EE resources that can be managed by setup are as follows:

(1) Resource adapter

Manage resource adapters using one of the following methods:

- **Deploy and use resource adapters as J2EE resource adapters.**
  Deploy a resource adapter imported into a J2EE server as a shared standalone module. The resource adapter will be available to all the J2EE applications running on that J2EE server. The resource adapters deployed on J2EE servers are called *J2EE resource adapters*.
  For details on the setup of J2EE resource adapters, see *1.2.2 Setup for resource adapters*.
- **Include resource adapters in J2EE applications.**
  You can use resource adapters included in a J2EE application in that J2EE application.
  For details on the applications of resource adapters included in J2EE applications, see *1.2.3 Setup for resource adapters included in J2EE applications*.

  Important note

  You can use any of the following resource adapters in J2EE applications. However, you cannot use resource adapters with the same display name simultaneously. If you specify the same display name, an error will occur.

The following table provides a comparison of the resource adapters used as J2EE resource adapters and the resource adapters included and used in J2EE applications.

Table 1–1: Types of resource adapters to be managed

Resource adapter			Used as a J2EE resource adapter	Included and used in a J2EE application
DB Connector	Transaction support level	NoTransaction or LocalTransaction	Y	Y
		XATransaction	Y	N
	Cluster configuration		Y	N
uCosminexus TP1 Connector	Transaction support level	NoTransaction or LocalTransaction	Y	Y
		XATransaction	Y	N
TP1/Message Queue - Access			Y	N
DB Connector for Cosminexus RM and Cosminexus RM			Y	N
Other resource adapters			Y	Y#

Legend:

Y: Can be used

N: Cannot be used

#

You cannot include and manage the following resource adapters in J2EE applications:

- Resource adapters with global transaction management (XATransaction) specified in the transaction support level
- Resource adapters containing a native library
- Resource adapters requiring control of starting order

(2) J2EE resources other than resource adapters

Manage the following J2EE resources in addition to the resource adapters:

- JavaBeans resources
- Mail configuration

For details on the setup of J2EE resources other than resource adapters, see *1.2.4 Setup for J2EE resources other than resource adapters*.

1.2.2 Setup for resource adapters

This subsection describes the setup of resource adapters deployed and used as J2EE resource adapters.

(1) Types of resource adapters

The types of resource adapters deployed and used as the J2EE resource adapters are as follows:

- DB Connector
  Use this resource adapter when you use the JDBC interface for connecting to a database.
- DB Connector for Cosminexus RM and Cosminexus RM
  Use this resource adapter when you use JMS interface for connecting to a database.
  For DB Connector for Cosminexus RM, see *2.7 Functions of DB Connector for Reliable Messaging*, and for the management of Cosminexus RM, see *2.3 Managing messages* in the manual *Cosminexus Reliable Messaging*.
- uCosminexus TP1 Connector
  Use this resource adapter for connecting to SPP of OpenTP1.
- TP1/Message Queue - Access
  Use this resource adapter for connecting to TP1/Message Queue.
- Other resource adapters
  You can use resource adapters compliant with Connector 1.5 for connecting to the optional resources.

(2) Flow of resource adapter management

The basic flow of resource adapter management is as follows:

1. Importing the resource adapters
2. Defining the resource adapter properties
3. Deploying the resource adapters

4. Testing the J2EE resource adapters for connectivity
5. Starting and stopping the J2EE resource adapters
6. Exporting the J2EE resource adapters

The description of above procedures is as follows:

- Importing the resource adapters
  You import the resource adapter.
- Defining the resource adapter properties
  You define the information required for connecting to a database.
- Deploying the resource adapters
  You deploy the imported resource adapter as a J2EE resource adapter. You can use the resource adapter deployed as the J2EE resource adapter from all the J2EE applications running on the J2EE server.
- Testing the J2EE resource adapters for connectivity
  You execute the connectivity test and confirm that the J2EE resource adapters are running correctly.
- Starting and stopping the J2EE resource adapters
  You start the J2EE resource adapters for which the setup is complete. Furthermore, stop the J2EE resource adapters according to the operations.
- Exporting the J2EE resource adapters
  You export the J2EE resource adapters according to the operations.
  To connect to a database, you must specify the database settings.
  Furthermore, if you want to use resource adapters, you must resolve the resource adapter references in J2EE applications. You resolve the resource adapter references by customizing J2EE applications. For an overview of customization of J2EE applications, see *1.3 Managing J2EE applications*.

(3) Setup of resource adapters

The following table describes the functions of the setup to be used in the resource adapter management.

Table 1–2: Setup used in resource adapter management

Operations in J2EE resource management	Functions	Commands
Import the resource adapters	Imports the resource adapters into the J2EE server.	`cjimportres` (`-type rar` specified)
Define the resource adapter properties	Obtains the pre-deployment resource adapter properties and generates a property file.	`cjgetresprop` (`-type rar` specified)
	Changes the values of the pre-deployment resource adapter properties to the values specified in the property file.	`cjsetresprop` (`-type rar` specified)
	Obtains the properties of the deployed J2EE resource adapters and generates the property file.	`cjgetrarprop`
	Changes the property values of the deployed J2EE resource adapters to the values specified in the property file.	`cjsetrarprop`
Deploy the resource adapters	Deploys the resource adapter.	`cjdeployrar`
Test the J2EE resource adapters for connectivity	Executes the connectivity test for the J2EE resource adapter.	`cjtestres` (`-type rar` specified)

Operations in J2EE resource management	Functions	Commands
Start and stop the J2EE resource adapters	Starts the J2EE resource adapter.	`cjstartrar`
	Stops the J2EE resource adapter that is running.	`cjstoprar`
Export the J2EE resource adapters	Exports the resource adapter residing on the J2EE server.	`cjexportrar`
Display the list of resources	Displays the list of the imported resource adapters.	`cjlistres` (`-type rar` specified)
	Displays the list of the J2EE resource adapters.	`cjlistrar`
Delete the resources	Deletes the resource adapters.	`cjdeleteres` (`-type rar` specified)
	Un-deploys the J2EE resource adapters.	`cjundeployrar`
Other operations	Displays the status of the resource adapter connection pool.	`cjlistpool`
	Deletes connections of the resource adapter connection pool.	`cjclearpool`
	Copies the resource adapter properties.	`cjcopyres` (`-type rar` specified)

1.2.3 Setup for resource adapters included in J2EE applications

This subsection describes the setup of resource adapters included in J2EE applications.

(1) Types of resource adapters

The types of resource adapters to be used by including in J2EE applications, and managed in the setup are as follows:

- DB Connector
  Use this resource adapter when you use the JDBC interface for connecting to the database.
  You cannot use this resource adapter in global transactions and the cluster configuration.
- uCosminexus TP1 Connector
  Use this resource adapter for connecting to SPP of OpenTP1.
  You cannot use this resource adapter in global transactions.
- Other resource adapters (resource adapters compliant with Connector 1.5)
  You can use resource adapters compliant with Connector 1.5 for connecting to optional resources.
  For details on the types of resource adapters that you can include and use in J2EE applications, see *3.3.2 Types of resource adapters* in the *uCosminexus Application Server Common Container Functionality Guide*.

(2) Flow of the resource adapter management

The basic flow for managing resource adapters included in J2EE applications is as follows:

1. Adding resource adapters to J2EE applications
2. Defining the properties of resource adapters included in J2EE applications
3. Testing the resource adapters included in J2EE applications for connectivity

4. Starting and stopping J2EE applications

5. Exporting J2EE applications

The description of each procedure is as follows:

- Adding resource adapters to J2EE applications
  Use one of the following methods for adding resource adapters in J2EE applications:
  - In the application development environment, create a J2EE application with the added resource adapter. Import that J2EE application into a J2EE server.
  - Import a resource adapter into the J2EE server, and then add the resource adapter to the J2EE application.
- Defining the properties of resource adapters included in J2EE applications
  Define the information required for connecting to a database.
- Testing resource adapters included in J2EE applications for connectivity
  Execute the connectivity test and confirm that the resource adapters included in the J2EE application are running correctly.
- Starting and stopping J2EE applications
  Start the J2EE application including the resource adapters for which the setup is complete. When the J2EE application starts, the resource adapters included in the J2EE application will start.
  Furthermore, stop the J2EE application according to the operations. When the J2EE application stops, the resource adapters included in the J2EE application will stop.
- Exporting J2EE applications
  Include resource adapters in the J2EE application and export the application as required in the operations.
  To connect to the database, you must specify the database settings.
  Furthermore, if you want to use resource adapters, you must resolve the resource adapter references in J2EE applications. You resolve the resource adapter references by customizing the J2EE application. For an overview of customization of J2EE applications, see *1.3 Managing J2EE applications*.

(3) Setup for resource adapters

The following table describes the functions of setup to be used in the management of resource adapters included in J2EE applications:

Table 1–3: Setup used in the management of resource adapters included in J2EE applications

Operations in J2EE resource management	Functions	Commands
Import resource adapters	Imports resource adapters into the J2EE server.	`cjimportres` (`-type rar` specified)
Add resource adapters to J2EE applications	Adds resource adapters to the J2EE application.	`cjaddapp` (`-type rar` specified)
Import J2EE applications	Imports J2EE applications including resource adapters.	`cjimportapp`
Define the resource adapter properties	Obtains properties of the resource adapters included in the J2EE applications and generates a property file.	`cjgetappprop` (`-type rar` specified)
	Changes the property values of the resource adapters included in J2EE applications to the values specified in the property file.	`cjsetappprop` (`-type rar` specified)

Operations in J2EE resource management	Functions	Commands
Test resource adapters for connectivity	Executes the connectivity test for the resource adapters included in J2EE applications.	`cjtestres` (`-name` [Application name] `-type rar` specified)
Start and stop J2EE applications	Starts the J2EE application. When the J2EE application starts, all the resource adapters included in the J2EE application start. You cannot start the resource adapters individually.	`cjstartapp`
	Stops the running J2EE application. When the J2EE application stops, all the resource adapters included in the J2EE application will stop. You cannot stop the resource adapters individually.	`cjstopapp`
Export J2EE applications	Includes resource adapters in J2EE applications and exports the resource adapters.	`cjexportapp`
Delete resource adapters from J2EE applications	Deletes resource adapters of J2EE resources from the J2EE applications.	`cjdeleteapp` (`-type rar` specified)
Display the list of resources	Displays the list of resource adapters included in J2EE applications.	`cjlistapp` (`-type rar` specified)
Other operations	Displays the connection pool status of the resource adapters included in J2EE applications.	`cjlistpool` (`-name` [Application name] specified)
	Deletes connections from the connection pools of the resource adapters included in J2EE applications.	`cjclearpool` (`-name` [Application name] specified)

1.2.4 Setup for J2EE resources other than resource adapters

This subsection describes the setup for J2EE resources other than resource adapters.

(1) Types of J2EE resources other than resource adapters

The types of J2EE resources other than resource adapters to be managed in the setup are as follows:

- JavaBeans resources
- Mail configuration

(2) Flow of J2EE resource management

The basic flow of J2EE resource management is as follows:

1. Importing J2EE resources
2. Defining J2EE resource properties
3. Testing J2EE resources for connectivity
4. Starting and stopping J2EE resources

In the management of J2EE resources using the server management commands, the required settings and operating methods depend on the types of J2EE resources.

The following table describes the operations required in each procedure, for each J2EE resource type. Execute the operations in the order of the numbers mentioned in the *Procedures* column.

Table 1–4: Operations required during J2EE resource management

Procedures		JavaBeans Resources	Mail Configuration
1.	Importing J2EE resources	Y	--
2.	Defining J2EE resource properties	Y	Y#
3.	Testing J2EE resources for connectivity	--	Y
4.	Starting and stopping J2EE resources	Y	--

Legend:

Y: Yes

--: Not applicable

#

Defining the mail configuration properties includes the creation of a new mail configuration.

The description of each procedure is as follows:

- Importing J2EE resources
  Import the JavaBeans resources.
- Defining J2EE resource properties
  Define the information required for managing JavaBeans resources and the mail configuration.
- Testing J2EE resources for connectivity
  Confirm that the mail configuration is running correctly.
- Starting and stopping J2EE resources
  Start JavaBeans resources for which the setup is complete. Furthermore, stop JavaBeans resources according to the operations.

(3) Setup for J2EE resources

The following table describes the functions for the setup to be used in the J2EE resource management:

Table 1–5: Setup used in J2EE resource management

Operations in J2EE resource management	Functions	Commands
Import J2EE resources	Imports JavaBeans resources into the J2EE server.	`cjimportjb`
Define J2EE resource properties	Obtains the HITACHI JavaBeans Resource Properties and generates the property file.	`cjgetjbprop`
	Changes the property values of the JavaBeans resources to the values specified in the property file.	`cjsetjbprop`
	Obtains the properties of the resources included in the mail and generates the property file.	`cjgetresprop` (`-type mail` specified)
	Changes the property values of the resources included in the mail to the values specified in the property file.	`cjsetresprop` (`-type mail` specified)
Test J2EE resources for connectivity	Executes the connectivity test for mails.	`cjtestres` (`-type mail` specified)
Start and stop J2EE resources	Start JavaBeans resources.	`cjstartjb`

Operations in J2EE resource management	Functions	Commands
	Stops JavaBeans resources.	`cjstopjb`
Display the list of resources	Displays the list of imported JavaBeans resources.	`cjlistjb`
	Displays the list of imported mails.	`cjlistres` (`-type mail` specified)
Delete the resources	Deletes JavaBeans resources.	`cjdeletejb`
	Deletes mails.	`cjdeleteres` (`-type mail` specified)
Other operations	Copies the mail property.	`cjcopyres` (`-type mail` specified)

1.3 Managing J2EE applications

This section provides an overview of J2EE application management in the setup.

1.3.1 J2EE applications to be managed

This subsection describes the format of J2EE applications to be managed and the components that configure J2EE applications.

(1) J2EE application format

The applications in one of the following formats are managed using the setup:

- **J2EE applications in the archive format**
  In this format, the configuration file of the J2EE application components is in the working directory of the J2EE server.
- **J2EE applications in the exploded archive format**
  In this format, the configuration file of the J2EE application components is not in the working directory of the J2EE server. This format uses the configuration files external to the J2EE server as logical J2EE applications.

For details on the J2EE application formats, see *15. Formats and Deployment of J2EE Applications* in the *uCosminexus Application Server Common Container Functionality Guide*.

(2) Components that configure J2EE applications

With the setup operations, you manage J2EE applications configured by the following components:

- **J2EE application consisting of servlets or JSPs**
- **J2EE application consisting of Enterprise Beans**
- **J2EE application consisting of servlets or JSPs and Enterprise Beans**

You can also include and manage resource adapters in J2EE application components.

When a J2EE application is in the archive format, you can use the components to make additions to the J2EE application and to create new J2EE applications. For details on creating J2EE applications, see *7. Creating J2EE Applications*.

1.3.2 Flow of J2EE application management

The flow of J2EE application management is as follows:

1. Importing the file
2. Creating the J2EE application#
3. Defining the J2EE application properties
4. Starting and stopping the J2EE application
5. Exporting the J2EE application

#

Do not execute this step in the case of J2EE applications that have already been created and set up by using the application development tool.

The following table describes the operations required for J2EE application management. You execute the operations in the order of the numbers mentioned in the *Procedures* column:

Table 1–6: Operations necessary for J2EE application management

Procedures		New J2EE application to be created	Already created J2EE application[#1]	
			Archive format	Exploded archive format
1.	Importing the file	Y[#2]	Y	Y
2.	Creating the J2EE application	Y	--[#3]	--
3.	Defining J2EE application properties	Y	Y	Y[#4]
4.	Starting and stopping the J2EE application	Y	Y	Y
5.	Exporting the J2EE application	Y	Y	Y

Legend:

Y: Yes

--: Not applicable

#1

This J2EE application is already created with an application development tool. The method of importing a J2EE application depends on the J2EE application format:

- When setting up a J2EE application in the archive format, specify the archive file path and import the J2EE application.
- When setting up a J2EE application in the exploded archive format, specify the application directory path or the exploded directory path and import the J2EE application.

#2

Import the following files for each component configuration:

- When creating a J2EE application that includes a servlet or JSP, import the servlet or JSP.
- When creating a J2EE application that includes an Enterprise Bean, import the Enterprise Bean.
- When creating a J2EE application that includes a resource adapter, import the resource adapter.

#3

In the created J2EE application in the archive format, you can add or delete the components or the library JAR.

#4

In the case of J2EE applications that are in the exploded archive format and are running on a host that is different from the host that executes the server management commands, you cannot set up or customize the J2EE application.

Each procedure is described below:

- Importing the file

  You import the components (servlets or JSPs, Enterprise Beans, resource adapters, and library JAR) that form the J2EE application or the created J2EE application.

- Creating the J2EE application
  You create the J2EE application with the imported file.
- Defining J2EE application properties
  Specify the property according to the components that configure J2EE applications, for the created J2EE application or imported and created J2EE applications.
  - Define properties corresponding to the DD statement
  - Set up the runtime properties and operations
  - Resolve J2EE resource references
- Starting and stopping the J2EE application
  You start the J2EE applications for which setup and customization is complete. Also, you stop applications as required by the operations.
- Exporting the J2EE application
  You export applications as required by the operations.

1.3.3 Setup used for managing the J2EE applications

The following table describes the setup functions that are used for managing the J2EE applications:

Table 1–7: Server management commands used for managing the J2EE applications

Operations in J2EE application management	Functions	Commands
Import the file	Imports the components (WAR files, EJB-JAR files, and RAR files) forming the J2EE application into the J2EE server.	`cjimportres`
	Imports the library JAR files into the J2EE server.	`cjimportlibjar`
	Imports the J2EE applications into the J2EE server.	`cjimportapp`
Create the J2EE application	Adds the imported EJB-JAR files, WAR files, and RAR files to the J2EE application.	`cjaddapp`
Define J2EE application properties	Acquires the attributes of the J2EE application or the components included in the J2EE application and generates the attribute file.	`cjgetappprop`
	Changes the attributes of the J2EE application or the components included in the J2EE application to the values of the specified application attribute file.	`cjsetappprop`
Start and stop the J2EE application	Starts the J2EE application so that the requests from the client can be received.	`cjstartapp`
	Stops the J2EE application so that the requests from the client are not received.	`cjstopapp`
Export the J2EE application	Exports the J2EE application that is on the J2EE server.	`cjexportapp`
Display the list of J2EE applications	Displays the names and status of all J2EE applications and a list of EJB-JAR files, WAR files, or RAR files included in the J2EE application.	`cjlistapp`
	Displays the list of library JAR files included in the J2EE application.	`cjlistlibjar`
	Displays the list of transaction information running on the J2EE server.	`cjlisttrn`

Operations in J2EE application management	Functions	Commands
	Displays the list with information on transactions that are being terminated on the J2EE server.	`cjlisttrnfile`
Delete the J2EE application	Deletes the J2EE applications or the EJB-JAR files, WAR files, and RAR files included in the J2EE application from the specified J2EE server.	`cjdeleteapp`
	Deletes the library JAR files from the J2EE application.	`cjdeletelibjar`
Other operations	Replaces the J2EE applications that are in the exploded archive format. #	`cjreloadapp`
	Replaces the J2EE applications that are in the archive format. #	`cjreplaceapp`
	Changes the names of the J2EE applications.	`cjrenameapp`
	Displays the status of the connection pool of the invoked resource adapters.	`cjlistpool`
	Generates the SQL statement for the CMP2.x Entity Bean.	`cjgencmpsql`
	Adds the user to the role.	`cjmapsec`
	Adds the user or role.	`cjaddsec`
	Deletes the user or role.	`cjdeletesec`
	Displays the list of users or roles.	`cjlistsec`
	Acquires the RMI-IIOP stubs and interfaces of the J2EE applications.	`cjgetstubsjar`

#

For details on the J2EE application format, see *1.3.1 J2EE applications to be managed*.

1.4 Setup restrictions

This section describes the restrictions depending on the runtime status of the host and the J2EE applications when setting up the J2EE application.

1.4.1 Restrictions on the host used for setup

You can execute the setup from a host that is different from the J2EE server and the CORBA Naming Service.

J2EE applications in the exploded archive format, however, can only be set up and customized on the same host as the J2EE server.

When operating a J2EE server on another host, the following settings need to be specified beforehand:

- Access permission settings for the J2EE server

  Access permissions are required for the J2EE server to be operated. In order to set the access permission for the J2EE server, define the webserver.connector.http.permitted.hosts key in the `usrconf.properties` file for the J2EE server. For details on the `usrconf.properties` file for the J2EE server, see *2.2.3 usrconf.properties (User property file for J2EE servers)* in the *uCosminexus Application Server Definition Reference Guide*.

  If there is no access permission for the J2EE server, an error message is displayed and the setup cannot be executed.

- Network settings

  You need to set up the network in advance so that the host that executes the server management commands can resolve the host name of the J2EE server.

If the setup, the J2EE server, and CORBA Naming Service are on different hosts, and if the Cosminexus version installed on these hosts is different, the setup may not run normally. Ensure that the same version of Cosminexus is installed on the hosts on which the setup, the J2EE server, and CORBA Naming Service are running.

1.4.2 Restrictions based on the J2EE application status

Depending on the status of the J2EE applications, some J2EE application operations can be performed and some operations cannot be performed.

The following table describes the status of J2EE applications:

Table 1–8: Status of J2EE applications

No.	Status#	Description
1	Started	The J2EE application has started.
2	Locking	The lock process is in process.
3	Locked	The lock process terminated normally.
4	Normal termination	Normal termination is in process.
5	Forced termination	Forced termination is taking place due to the forced termination operation.
6	Terminated	The J2EE application ended.
7	Lock failure	The lock process ended abnormally.

No.	Status#	Description
8	Normal termination failure	The normal termination process ended abnormally.
9	Forced termination failure	The forced termination process ended abnormally.

#

For more details on how to reference the J2EE application status, see *10.3 Referencing the list of J2EE applications*.

The following table describes whether the operations can be performed for each of the above-mentioned status of J2EE applications:

Table 1–9: Status of the J2EE applications and the operations that can be performed

Operations (Commands)		Status of the J2EE applications								
		1	2	3	4	5	6	7	8	9
Import the file	Import the WAR files, the EJB-JAR files, and the RAR files (`cjimportres`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Import the library JAR files (`cjimportlibjar`)	--	--	--	--	--	Y	--	--	--
Create the J2EE application (`cjaddapp`)		--	--	--	--	--	Y	--	--	--
Set and customize the J2EE applications	Acquire the attribute files (`cjgetappprop`)#1	Y	--	--	--	--	Y	--	--	--
	Set up the attributes (`cjsetappprop`)#1	--	--	--	--	--	Y	--	--	--
Start and stop the J2EE applications	Start the J2EE applications (`cjstartapp`)	--	--	--	--	--	Y	--	--	--
	Normal termination of the J2EE applications (`cjstopapp`)	Y	--	--	--	--	--	--	--	--
	Manual forced termination of the J2EE applications (`cjstopapp`)	--	--	Y	Y	--	--	--	--	--
	Automatic forced termination of the J2EE applications (`cjstopapp`)	Y	--	--	--	--	--	--	--	--
Export the J2EE applications (`cjexportapp`)		Y	--	--	--	--	Y	--	--	--
Display the list of J2EE applications	Display the status of J2EE applications (`cjlistapp`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Display the list of library JAR (`cjlistlibjar`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Display the list of transaction information being operated (`cjlisttrn`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Display the list of transaction information being terminated (`cjlisttrnfile`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
Delete the J2EE applications	Delete the J2EE applications and the configuration components (`cjdeleteapp`)	--	--	--	--	--	Y	--	--	--

Operations (Commands)		Status of the J2EE applications								
		1	2	3	4	5	6	7	8	9
	Delete the library JAR (`cjdeletelibjar`)	--	--	--	--	--	Y	--	--	--
Other operations	Replace the J2EE applications in the exploded archive format (`cjreloadapp`)	Y	--	--	--	--	Y	--	--	--
	Replace the J2EE applications in the archive format (`cjreplaceapp`)	Y	--	--	--	--	Y	--	--	--
	Change the name of the J2EE applications (`cjrenameapp`)	--	--	--	--	--	Y	--	--	--
	Display the status of connection pool of the resource adapter (`cjlistpool`)	Y	--	--	--	--	--	--	--	--
	Generate the SQL statements (`cjgencmpsql`)	--	--	--	--	--	Y	--	--	--
	Register the user in the role (`cjmapsec`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Add the user or role (`cjaddsec`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Delete the user or role (`cjdeletesec`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Display the list of users or roles (`cjlistsec`)	Y	Y	Y	Y	Y	Y	Y	Y	Y
	Acquire the RMI-IIOP stubs and interfaces (`cjgetstubsjar`)	Y	Y	Y	Y	Y	Y#2	Y	Y	Y

Legend:

Numbers: The status of the J2EE applications corresponding to the serial numbers in Table 1-8

Y: Can be executed

--: Cannot be executed

#1

When customizing the J2EE application and changing the properties of each component, you can get the attribute file even if the J2EE application has started. In order to apply the values of the edited attribute file in the changed contents, stop the J2EE application.

#2

If the J2EE application is not yet started, the operation cannot be performed.

1.4.3 Other restrictions

When a server management command is executed while a different server management command is processing, the process execution on each server is controlled based on the *type of server management commands* that are processing. For more details on the restrictions when server management commands are concurrently executed, see *3.2 Exclusive access control of server management commands*.

2 Interfaces Used for Application Setup

This chapter gives an overview of the server management commands that are the interfaces used for application setup.

2.1 Functionality of server management commands

For application setup, server management commands are provided as interfaces. The server management commands are a group of commands for setting and managing the resources and J2EE applications operating on a J2EE server.

The following table describes the functionality of the server management commands used for application setup.

Table 2–1: List of functionality of server management commands

Functionality		Reference location
Setting resource adapters	Settings for connecting to the database	*4.2*
	Settings for connecting to other resources	*4.3*
Setting up resource adapters included in J2EE applications		*Chapter 5*
Settings for J2EE resources other than resource adapters	Settings for JavaBeans resources	*6.2*
	Settings for mail configuration	*6.3*
Creating J2EE applications (EAR)		*Chapter 7*
Importing#1 and exporting J2EE applications		*Chapter 8*
Property settings of J2EE applications		*Chapter 9*
Executing J2EE applications	Starting and stopping J2EE applications	*10.2*
	Referencing the list of J2EE applications	*10.3*
	Deleting J2EE applications	*10.4*
	Switching between J2EE applications#2	*10.5*
	Changing the J2EE application name	*10.6*
	Acquiring the RMI-IIOP stubs and interfaces	*10.7*
	Referencing the list of transactions	*10.8*

#1

You can import J2EE applications having the following formats:

- J2EE applications in the archive format (EAR files)
- J2EE applications in the exploded archive format (application directory)

#2

When replacing the deployed J2EE applications, the operations differ as follows, based on the format of J2EE applications while importing the J2EE applications:

- When replacing the deployed J2EE applications having the archive format, you must restart (stop, replace, and start) the J2EE applications once.
- When replacing J2EE applications having the exploded archive format, you can execute the server management commands for replacing the J2EE applications without restarting (stopping, replacing, and starting) the J2EE applications that are already running (reload functionality).

For details on replacing J2EE applications, see *5.6 Switching the J2EE Application* in the *uCosminexus Application Server Operation, Monitoring, and Linkage Guide*.

3 Basic Operations of Server Management Commands

This chapter describes the basic operations when using the server management commands to set up an application.

3.1 Prerequisites for using the server management commands

Confirm the following items when using the server management commands:

- The server management commands are commands that can be executed while the J2EE server is running. When using the server management command, execute the command after starting the J2EE server and the corresponding prerequisite processes. For details on starting the J2EE server and the corresponding prerequisite processes, see *2.3 Mechanism of Starting and Stopping the Logical Server* in the *uCosminexus Application Server Operation, Monitoring, and Linkage Guide*.
- The root permission or the Component Container Administrator permission is required for executing a server management command. Set the permission required for the execution.
- The administrator privileges are required for executing Cosminexus commands in Windows. For details on the precautions that you must take when using the Cosminexus commands in Windows, see *1.4 Precautions for using commands* in the *uCosminexus Application Server Command Reference Guide*.

For details on the precautions that you must take when using the server management commands, see *1.4 Precautions for using commands* in the *uCosminexus Application Server Command Reference Guide*.

3.2 Exclusive access control of server management commands

This section describes the system of server management commands and the exclusive access control of server management commands.

3.2.1 Types of server management commands

When a server management command is executed while a different server management command is processing, Cosminexus controls the process execution on each server based on the *type of server management commands* that are processing.

Server management commands are classified into three types as follows:

- **Reference node commands**
  These server management commands display the configuration status of a J2EE server such as the number of imported J2EE applications and their status, and the name of the resources included. This command does not update the contents of a J2EE server.
- **Update node commands**
  These server management commands update the contents of a J2EE server and acquire configuration information.
- **Privilege node commands**
  These server management commands update the contents of a J2EE server and are always given priority for executing a process as compared to the other commands.

For details on each type of server management command, see the *uCosminexus Application Server Command Reference Guide*.

3.2.2 Exclusive access control of server management commands

The following table describes the conditions for exclusive access control of command execution for each type of server management command.

Table 3–1: Conditions for exclusive access control of command execution

Running command	Command to be executed later		
	Update node command	Reference node command	Privilege node command
Update node command	AF	P	P
Reference node command	P	AF	P
Privilege node command	P	P	C

Legend:
P: Command execution is permitted.
AF: Command execution will start after the running command finishes.
C: Command execution is cancelled. An exclusive error is returned.

If you attempt to execute another server management command while other commands are running, Cosminexus controls the server management command to be executed in a different format depending on the type of the command to be executed.

(1) Control for executing a server management command in a J2EE server where another server management command is running

When a server management command of a different type is executed for the J2EE server in which a server management command is being processed, processing of the respective server management commands is executed concurrently. When a server management command is being processed on a J2EE server and another server management command of the same type is concurrently executed, the processing request of the server management command executed later will go into waiting status in the sequence of execution, and the processing will start after the processing of the previously executed server management command finishes. Therefore, for server management commands of the same type, the number of server management commands that can be concurrently executed is one. There is no limit to the maximum number of server management commands in the waiting status. If a process execution is in the waiting status, the message KDJE42211-I is displayed on the J2EE server machine.

(2) Control when using the server management commands while the shutdown command is being processed

Server management commands classified as update node and reference node cannot be executed for a J2EE server in which the `shutdown` command is being processed or is pending. When the server management commands classified as reference node and update node are executed during the shutdown process, an exclusive error occurs and the message KDJE42212-E is displayed. Note that server management commands classified as privilege node can be executed.

(3) Control when using the privilege node commands

Privilege node commands can be executed with priority over all other commands. Even when a privilege node command is being executed, it does not affect the execution of other commands, however, only one privilege node command can be executed concurrently for one J2EE server. When two or more privilege node commands are executed for one J2EE server, an error occurs and the message KDJE42230-E is displayed.

3.2.3 Forced release of exclusive access control of server management commands

When the same exclusive error message (information about the command being executed is the same) is displayed even if a server management command is executed several times at short intervals, there may be some discrepancy in the exclusive information of the server management command due to the abnormal termination of the server management command. In this case, confirm that the command displayed in the message is not running, forcibly release the exclusive access control of the server management command, and then reset the exclusive information.

You need to specify the J2EE server to forcibly release the exclusive access control.

Note:

If the exclusive access control is released when a server management command is being executed, multiple server management commands are executed concurrently and some discrepancy may arise in the J2EE server. Therefore, ensure that no exclusive information is erroneously released when these server management commands are executed normally.

The procedure for forcibly releasing the exclusive access control of a server management command is as follows:

1. Before forcibly releasing the exclusive access control, check that the server management command is not running.

   If you try to execute the server management command, message KDJE37057-E or message KDJE37301-E is displayed. Check that the server management command displayed in these messages is not running.

2. Execute the `cjresetsv` command to forcibly release the exclusive access control.

   Execution format

   ```
   cjresetsv [J2EE-server-name] [-nameserver provider-URL]
   ```

   Execution example

   ```
   cjresetsv MyServer
   ```

3.3 Customizing operation settings of server management commands

If you edit a user-defined file, you can customize the operation settings of server management commands. If necessary, customize the operation settings of server management commands. Note that in UNIX, you need either the root authority or the authority of the Component Container Administrator for performing customization.

The commands in the following directories are the server management commands:

- In Windows
  Cosminexus-installation-directory`\CC\admin\bin`
- In UNIX
  `/opt/Cosminexus/CC/admin/bin`

For details on the commands, see the *uCosminexus Application Server Command Reference Guide*.

3.3.1 User-defined files to be customized

To customize the operation settings of server management commands, edit the following user-defined files in a text editor:

- **usrconf.properties (system property file for server management commands)**
  This file specifies the properties of server management commands.
  The storage location of the file is as follows:
  - **In Windows**
    Cosminexus-installation-directory`\CC\admin\usrconf\usrconf.properties`
  - **In UNIX**
    `/opt/Cosminexus/CC/admin/usrconf/usrconf.properties`

  For details, see *5.2.3 usrconf.properties (System property file for server management commands)* in the *uCosminexus Application Server Definition Reference Guide*.
- **usrconf.bat (option definition file for server management commands)**
  This file specifies the JavaVM startup options for server management commands in Windows.
  The storage location of the file is as follows:
  Cosminexus-installation-directory`\CC\admin\usrconf\usrconf.bat`
  For details, see *5.2.2 usrconf.bat (Option definition file for server management commands for Windows)* in the *uCosminexus Application Server Definition Reference Guide*.
- **usrconf (option definition file for server management commands)**
  This file specifies the JavaVM startup options for server management commands in UNIX.
  The storage location of the file is as follows:
  `/opt/Cosminexus/CC/admin/usrconf/usrconf`
  For details, see *5.2.1 usrconf (Option definition file for server management commands for UNIX)* in the *uCosminexus Application Server Definition Reference Guide*.

3.3.2 Main items that can be set up during the customization of server management commands

Of the items that can be set up by editing the user-defined files during the customization of server management commands, the following table describes the main items, the user-defined files in which the main items are set up, and the keys used to set up the main items. For details on the keys and for information on the keys that are not described here, see *5. Files Used in Server Management Commands* in the *uCosminexus Application Server Definition Reference Guide*.

Table 3–2: Main items that can be set up during customization of server management commands

Classification	Item	Setup file	Setup method
JavaVM	JavaVM heap size	`usrconf.bat` (in Windows) or `usrconf` (in UNIX)	Specify the heap size of JavaVM in the `add.jvm.arg` key.
Smart Agent	--	`usrconf.properties`	In the `vbroker.agent.port` key, specify the port number of the Smart Agent that the server management commands use.
Log	Trace file of Cosminexus TPBroker	`usrconf.properties`	In the `vbroker.orb.htc.tracePath` key, specify the path for the output destination of the trace file of Cosminexus TPBroker.
	Log of server management commands	• `usrconf.bat` (in Windows) or `usrconf` (in UNIX) • `usrconf.properties`	You can change the log output destination and log level of the log for server management commands. For details, see *3.4 Settings for log acquisition of server management commands*.

Legend:

--: Not applicable.

Note

Customization of server management commands is necessary even when the functionality of naming management is used. For details, see *2.3.5 Settings in the execution environment* in the *uCosminexus Application Server Common Container Functionality Guide*.

3.4 Settings for log acquisition of server management commands

You can change the log output destination and log level for the logs of server management commands. The items that you can change, and the user-defined files and keys corresponding to the items are as follows:

Item	Corresponding user-defined file and key
Log output destination	`-Dejbserver.log.directory` option of the `USRCONF_JVM_ARGS` key of `usrconf.bat` (in Windows) or `usrconf` (in UNIX)
Log level	`ejbserver.logger.enabled.*` key of `usrconf.properties`

For details on the files and keys, see *5. Files Used in Server Management Commands* in the *uCosminexus Application Server Definition Reference Guide*.

Important note

The disk space required for the log output directory of the server management commands varies according to the value specified in the `ejbserver.cui.logfile.compatible` key.

When the specification value is `true`

51,655KB + trace information of TPBroker

When the specification value is `false`

18,624KB + trace information of TPBroker

For details on the disk space required to output the trace information of TPBroker, see the description related to the occupied disk space in the *TPBroker Operation Guide*.

3.4.1 Changing the log output destination

To change the log output destination of server management commands, specify the output-destination directory of logs in `usrconf.bat` (in Windows) or `usrconf` (in UNIX) for server management commands.

Changing the log output destination

The default log output destination is as follows:

- In Windows
  Cosminexus-installation-directory`\CC\admin\logs`
- In UNIX
  `/opt/Cosminexus/CC/admin/logs/`

You can change the output destination for the logs of server management commands with the `-Dejbserver.log.directory` option of the `USRCONF_JVM_ARGS` key of `usrconf.bat` (in Windows) or `usrconf` (in UNIX).

Setting example

- In Windows
  ```
  set USRCONF_JVM_ARGS="-Dejbserver.log.directory=C:\CClogs\admin"
  ```
- In UNIX

```
set USRCONF_JVM_ARGS="-Dejbserver.log.directory=/CClogs/admin"
```

Current directory

The current directory when the log output destination is specified by a relative path is the current directory when a server management command is executed.

For example, when the user ID is `user1` and the home directory of the user is `C:\Documents and Settings\user1`, and you log in with this user ID, the current directory immediately after login becomes `C:\Documents and Settings\user1`. Here, if you execute a server management command, the current directory becomes `C:\Documents and Settings\user1`.

Important note

- When you set the output destination of Java VM maintenance information and GC log data, note that the specification in the `usrconf.cfg` file (`add.jvm.arg=-XX:HitachiJavaLog:`*destination*) prevails over the specification in the user definition file.
- When you perform operations with the Management Server remote management function, you cannot change the log output destination of server management commands.

3.4.2 Changing the log level

The log level of server management commands represents the importance of logs. There are four log levels; `Error`, `Warning`, `Information`, and `Debug`. If you set a log level, the log of the set level is output. By default, only the log of the `Error` level is acquired. In normal cases, use the default settings.

A log level is set by the following key of `usrconf.properties` for the server management commands:

`ejbserver.logger.enabled.*=`*level-name*

In the level name, specify either one or multiple character strings from `Error`, `Warning`, `Information`, and `Debug`. To specify multiple levels, demarcate the character strings of the level name with a comma (`,`).

Coding examples:

1. `ejbserver.logger.enabled.*=Error`
2. `ejbserver.logger.enabled.*=Error,Warning`
3. `ejbserver.logger.enabled.*=Error,Warning,Information`
4. `ejbserver.logger.enabled.*=Error,Warning,Information,Debug`

Important note

- The number of logs that can be acquired increases in the order of 1, 2, 3, and 4 of the coding examples. If you set multiple log levels and acquire the logs, performance deteriorates, and switching of the number of log files occurs frequently.
- When you specify a character string other than `Error`, `Warning`, `Information`, or `Debug` in the level name, or when you specify a blank value, the `KDJE90009-W` message is output. The log of the `Error` level is acquired.

3.5 Property settings using the property file

With the server management commands, you use property files to set up the properties of J2EE resources and J2EE applications.

For details on how to set up the properties using `cosminexus.xml`, see *13.2 Managing properties* in the *uCosminexus Application Server Common Container Functionality Guide*.

3.5.1 Procedure for setting the properties of a J2EE resource adapter

This subsection describes the procedure for setting up the properties of resource adapters and other J2EE resources.

(1) Procedure for setting up the properties of resource adapters

The procedure for setting up the properties of a resource adapter deployed and used as a J2EE resource adapter is described here.

For details on the procedure to set up the properties of resource adapters included in J2EE applications, see *3.5.2 Procedure for setting the properties of a J2EE application*.

1. Acquire the attribute file.
   When acquiring the attribute file of an un-deployed resource adapter, execute the `cjgetresprop` command. When acquiring the attribute file of the resource adapter after deploying the resource adapter, execute the `cjgetrarprop` command.
   You must acquire the Connector property file.
   For details on the `cjgetresprop` command, see *cjgetresprop (get resource Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjgetrarprop` command, see *cjgetrarprop (get RAR file Property)* in the *uCosminexus Application Server Command Reference Guide*.

2. Edit the property settings.
   Edit the acquired attribute file by using a text editor. For editing the items in the attribute file, see the explanation about each property setting.
   For details on the property file, see *4.1 HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

3. Set the properties.
   When setting the properties of an un-deployed resource adapter, execute the `cjsetresprop` command. When setting the properties of a deployed resource adapter, execute the `cjsetrarprop` command.
   For details on the `cjsetresprop` command, see *cjsetresprop (set Property of resource)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetrarprop` command, see *cjsetrarprop (HITACHI Connector Property Settings)* in the *uCosminexus Application Server Command Reference Guide*.

(2) Procedure for setting up the properties of J2EE resources other than resource adapters

The procedure for setting up the properties of JavaBeans resources, and the mail configuration is as follows:

1. Acquiring the property file.

For acquiring the property file of JavaBeans resources, execute the `cjgetjbprop` command. For acquiring the property file of the mail configuration, execute the `cjgetresprop` command.

The property files to be acquired are as follows:

- For a JavaBeans resource
  JavaBeans resource property file
- For a mail configuration
  Mail property file

For details on the `cjgetjbprop` command, see *cjgetjbprop (get HITACHI JavaBeans Resource Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjgetresprop` command, see *cjgetresprop (get resource Property)* in the *uCosminexus Application Server Command Reference Guide*.

2. Editing the property settings.
   You use a text editor to edit the acquired property file. For editing the items in the property file, see the explanation of each property setting.
   For details on the property file, see *2.2.9 Details of the Connector property* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

3. Setting up the properties.
   For setting up the JavaBeans resource properties, execute the `cjsetjbprop` command. For setting up the mail configuration properties, execute the `cjsetresprop` command.
   For details on the `cjsetjbprop` command, see *cjsetjbprop (set HITACHI JavaBeans Resource Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetresprop` command, see *cjsetresprop (set Property of resource)* in the *uCosminexus Application Server Command Reference Guide*.

3.5.2 Procedure for setting the properties of a J2EE application

The procedure for setting the properties of a J2EE application are as follows:

1. Acquire the attribute file.
   For acquiring the property file of the application properties and the properties of the components (servlets or JSPs, Enterprise Beans, and resource adapters) that configure J2EE applications, you execute the `cjgetappprop` command.
   The property file you must acquire depends on the component for which the properties are to be setup. Furthermore, you acquire the application integrated property file for editing the attribute information of the components of the application in a batch.
   For details on the `cjgetappprop` command, see *cjgetappprop (get HITACHI Application Property)* in the *uCosminexus Application Server Command Reference Guide*.

2. Edit the property settings.
   Edit the acquired attribute file by using a text editor. For editing the items in the attribute file, see the explanation about each property setting.
   For details on property files, see *3. Property Files Used for Setting J2EE Applications* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

3. Set the properties.
   For setting up the property file of the application properties and the properties of the components (servlets or JSPs, Enterprise Beans, and resource adapters) that configure a J2EE application, execute the `cjsetappprop` command.

For details on the `cjsetappprop` command, see *cjsetappprop (set HITACHI Application Property)* in the *uCosminexus Application Server Command Reference Guide*.

Reference note

For creating a new J2EE application, add the components (servlets or JSPs, Enterprise Beans, and resource adapters that configure a J2EE application) to the J2EE application, and before adding the components use the `cjgetresprop` and `cjsetresprop` commands for defining the properties of the components. For details on the `cjgetresprop` command, see *cjgetresprop (get resource Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetresprop` command, see *cjsetresprop (set Property of resource)* in the *uCosminexus Application Server Command Reference Guide*.

3.6 Provider URL specified using the server management commands

In the case of server management commands, you can specify the value called *Provider-URL* in options.

When a server management command is executed by omitting the *Provider-URL*, the value set in the `usrconf.properties` file is assigned by default.

The `usrconf.properties` file is saved in the following directories:

In Windows

```
Cosminexus-installation-directory\CC\admin\usrconf\usrconf.properties
```

In UNIX

```
/opt/Cosminexus/CC/admin/usrconf/usrconf.properties
```

Examples of executing a server management command using the value set in the `usrconf.properties` file and of specifying the provider URL with the `cjlistapp` command are as follows:

An example of executing a server management command with the value set in the usrconf.properties file:

In Windows

```
Cosminexus-installation-directory\CC\admin\bin\cjlistapp [server-name]
```

In UNIX

```
/opt/Cosminexus/CC/admin/bin/cjlistapp [server-name]
```

An example of executing a server management command by specifying the provider URL:

In Windows

```
Cosminexus-installation-directory\CC\admin\bin\cjlistapp [server-name]
-nameserver Provider-URL
```

In UNIX

```
/opt/Cosminexus/CC/admin/bin/cjlistapp [server-name] -nameserver Provid
er-URL
```

Specify the provider URL in the following format:

Execution format

```
Protocol-name::Host-name:Port-number
```

Protocol-name

Protocol-name is the protocol name for the CORBA naming service. It is fixed as corbaname. When iioploc or iiopname is specified, the protocol name can be converted into corbaname.

Host-name

Host-name is the name of the host on which the CORBA naming service is running.

Port-number

Port-number is the number of the port on which the CORBA naming service is running.

The exclusive information of the server management command of the CORBA naming service specified in the provider URL is released.

3.7 Description of server management commands in this manual

This manual describes the use of server management commands as follows:

- The execution format and the example of execution describe only the arguments mainly related to the operation. For details on the commands, such as specifiable arguments, see the *uCosminexus Application Server Command Reference Guide*.
- Only the name of the command to be executed is described. When specifying a path for executing the commands, specify the path containing the storage location of the command.

  The commands are stored in the following directories:

  In Windows

  Cosminexus-installation-directory`\CC\admin\bin\`

  In UNIX

  `/opt/Cosminexus/CC/admin/bin/`

4 Setting Resource Adapter

This chapter describes the setup of resource adapters to be deployed and used as J2EE resource adapters. A J2EE resource adapter is a resource adapter that is deployed as a shared standalone module on a J2EE server.

For details on the setup of resource adapters included in J2EE applications, see *5. Setting up Resource Adapters Included in J2EE Applications*.

4.1 Overview of Resource adapter settings

The resource adapter settings include the operations that enable the usage of resource adapters from J2EE applications.

This section describes the types of resource adapters that you can deploy and use as J2EE resource adapters, and gives an overview of the setup.

4.1.1 Available resource adapters

You can deploy and use the following resource adapters as J2EE resource adapters:

- DB Connector
- DB Connector for Cosminexus RM and Cosminexus RM
- uCosminexus TP1 Connector
- TP1/Message Queue - Access
- Other resource adapters compliant with Connector 1.0 or Connector 1.5#

#

For the types of other resource adapters that are compliant with Connector 1.0 or the resource adapters compliant with Connector 1.5, see *3.3.2 Types of resource adapters* in the *uCosminexus Application Server Common Container Functionality Guide*.

Important note

You can use J2EE resource adapters and the resource adapters included in J2EE applications simultaneously from J2EE applications. However, you cannot use J2EE resource adapters and the resource adapters included in the J2EE application with the same display name simultaneously. If you specify the same display name, an error will occur.

4.1.2 Overview of settings and operations

This subsection gives an overview for the following resource adapter settings:

- Settings for connecting to the database
- Settings for connecting to other resources (when the resource adapter is used)
- Common settings for resource adapters

For details on the settings for connecting to a database queue, see the manual *Cosminexus Reliable Messaging*.

(1) Settings for connecting to the database

This operation is necessary when using the DB Connector for connecting to the database.

Table 4–1: Overview of settings required for connecting to the database

Settings	Contents	Reference
Importing the DB Connector	Import the RAR file of the DB Connector and add the file to the resources that can be used from the J2EE server.	*4.2.1*
Defining the DB Connector properties	Set the following information for the database connection: • General information about the DB Connector • Configuration properties • Runtime properties • Optional name information	*4.2.2*
Deploying the DB Connector	Deploy the DB Connector on the basis of the imported RAR file. When deployed, the DB Connector can be used as a J2EE resource adapter.	*4.2.3*
Testing the connectivity of DB Connector	Verify that the contents set in the DB Connector are correct.	*4.2.4*
Starting the DB Connector	Start the DB Connector.	*4.2.5*
Stopping the DB Connector	Stop the DB Connector.	*4.2.6*
Undeploying the DB Connector	Delete the deployed DB Connector. Execute this setting as required when switching resource adapters.	*4.2.7*
Exporting the DB Connector	Export the J2EE resource adapter as a RAR file. Execute this setting when required.	*4.2.8*

(2) Settings for connecting to other resources (when the resource adapter is used)

This operation is required when using a resource adapter to connect to various resources like OpenTP1.

Table 4–2: Overview of settings required for connecting to other resources (when the resource adapter is used)

Settings	Contents	Reference
Importing the resource adapter	Import the resource adapter (RAR file) and add the resource adapter to the resources that can be used from the J2EE server.	*4.3.1*
Defining the resource adapter properties	Set the following information for connecting to the database: • General information of the resource adapter • Configuration properties • Runtime properties • Optional name information	*4.3.2* or *4.3.3*
Deploying the resource adapter	Deploy the resource adapter on the basis of the imported resource adapter. When deployed, the resource adapter can be used as a J2EE resource adapter.	*4.3.4*
Testing the connectivity of a J2EE resource adapter	Verify whether the contents set in the resource adapter are correct.	*4.3.5*
Starting the J2EE resource adapter	Start the J2EE resource adapter.	*4.3.6*
Stopping the J2EE resource adapter	Stop the J2EE resource adapter.	*4.3.7*
Undeploying the J2EE resource adapter	Delete the J2EE resource adapter. Execute this setting as required when switching resource adapters.	*4.3.8*
Exporting the J2EE resource adapter	Export the J2EE resource adapter as a RAR file.	*4.3.9*

Settings	Contents	Reference
	Execute this setting when required.	

(3) Common settings for resource adapters

Common settings refer to the settings that are common to the resource adapters.

Table 4–3: Overview of the common settings for resource adapters

Settings	Contents	Reference
Displaying the state and list of J2EE resource adapters	Reference the state of deployed J2EE resource adapters and the list of connection definition identifiers.	*4.4*
Displaying the list of resource adapters	Reference the list of imported resource adapters and the list of connection definition identifiers.	*4.5*
Checking the connection pool state	Reference the state of the resource adapter connection pool. Furthermore, delete the connections within the connection pool, as and when required.	*4.6*
Referencing and changing the resource adapter names registered in the JNDI name space	Reference the resource adapter names registered in the JNDI name space and specify an optional name, as and when required.	*4.7*
Delete the resource adapter	Delete the resource adapters.	*4.8*
Copy the resource adapter	The resource adapters can be copied to other folders.	*4.9*

4.2 Settings for connecting to the database

To connect to the database, you set up the DB Connector that is a resource adapter for connecting to the database.

The connection to the database with a DB Connector refers to the process of accessing only the database table using the JDBC interface.

To access a queue by using the JMS interface, use the DB Connector for Cosminexus RM and Cosminexus RM to connect to the database. For details, see the manual *Cosminexus Reliable Messaging*.

To set up the DB Connector, first set up the environment for the database to be used. Also, you must set up HiRDB Type4 JDBC Driver, Oracle JDBC Thin Driver, or SQL Server JDBC Driver depending on the types of DB Connector to be used.

Set up the DB Connector with the following procedure:

1. Import the DB Connector.
2. Define the properties.
3. Deploy the DB Connector.
   Deploying refers to the process of deploying the DB connector as a shared and standalone module (J2EE resource adapter) on the J2EE server.
4. Check the connectivity.
   You can confirm if the connection is correct with the help of a connectivity test.

For the deployed DB Connector, you need to resolve the resource adapter references by using the J2EE application property settings. For details, see *9.3.3 Defining resource adapter references*.

Reference note

To set up new DB Connector properties, you can use the template file provided in Cosminexus Component Container.

The template file of the HITACHI Connector Property file is stored in the following directory:

- In Windows:
  Cosminexus-installation-directory`\CC\admin\templates\`
- In UNIX:
  `/opt/Cosminexus/CC/admin/templates/`

You can use this template file to edit the HITACHI Connector Property file before importing the DB Connector. Copy and use the template file.

For details on the template file names of the HITACHI Connector Property file, see *4.1.13 Template files of the HITACHI Connector Property File* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

You do not use the template file when you change the properties of DB Connector for which properties are already specified. You obtain the Connector properties of an imported DB Connector, and then edit the HITACHI Connector Property file.

4.2.1 Importing the DB Connector

To import the DB connector, execute the following command:

(1) Execution format

```
cjimportres [server-name] [-nameserver provider-URL] -type rar -f file-path
```

Specify the RAR file in *file- path*.

The RAR file is saved in the following directory:

- In Windows:
  Cosminexus-installation-directory`\CC\DBConnector\`
- In UNIX:
  `/opt/Cosminexus/CC/DBConnector/`

When you import the DB Connector as a resource adapter, specify one of the following RAR files based on the transaction management method and the type of JDBC driver that will be used. Note that you cannot use a global transaction for connecting to XDM/RD E2 or SQL Server.

- **DBConnector_HiRDB_Type4_CP.rar**
  DBConnector_HiRDB_Type4_CP.rar is a DB Connector for HiRDB Type4 JDBC Driver. Choose this RAR file when you use a local transaction or when you do not use transaction management (when LocalTransaction or NoTransaction is specified in the transaction support level).
  Connect to HiRDB and XDM/RD E2 using the ConnectionPoolDataSource of HiRDBType4 JDBC Driver.
- **DBConnector_HiRDB_Type4_XA.rar**
  DBConnector_HiRDB_Type_XA.rar is a DB Connector for HiRDB Type4 JDBC Driver. Choose this RAR file when you use a global transaction (when XATransaction is specified in the transaction support level).
  Connect to HiRDB by using XADataSource of HiRDB Type4 JDBC Driver.
- **DBConnector_Oracle_CP.rar**
  The DBConnector_Oracle_CP.rar is a DB Connector for Oracle JDBC Thin Driver. Choose this RAR file when you use a local transaction or when you do not use transaction management (when LocalTransaction or NoTransaction is specified in the transaction support level).
  Connect to Oracle by using ConnectionPoolDataSource of Oracle JDBC Thin Driver.
- **DBConnector_Oracle_XA.rar**
  The DBConnector_Oracle_XA.rar is a DB Connector for Oracle JDBC Thin Driver. Choose this RAR file when you use a global transaction (when XATransaction is specified in the transaction support level).
  Connect to Oracle by using XADataSource of Oracle JDBC Thin Driver.
- **DBConnector_SQLServer_CP.rar**
  The DBConnector_SQLServer_CP.rar is a DB Connector for SQL Server JDBC Driver. Choose this RAR file when you use a local transaction or when you do not use transaction management (when LocalTransaction or NoTransaction is specified in the transaction support level).
  Connect to SQL Server by using ConnectionPoolDataSource of SQL Server JDBC Driver.

(2) Example of execution

```
cjimportres MyServer -type rar -f DBConnector_HiRDB_Type4_CP.rar
```

For details on the `cjimportres` command, see *cjimportres (import resource)* in the *uCosminexus Application Server Command Reference Guide*.

4.2.2 Defining the DB Connector properties

You define the DB Connector properties. You can also define the DB Connector properties after deploying the DB Connector. Note that if you want to change the properties of a DB Connector that is already deployed, change the properties when the applicable DB Connector is not running.

For details on property settings, see *3.5 Property settings using the property file*. The following subsections define the DB Connector properties:

(1) Attribute file to be edited

Connector attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire the Connector attribute file of DB Connector:
  Execution format
  ```
  cjgetresprop server-name [-nameserver provider-URL] -type rar -resname
  DB-Connector-display-name -c Path-of-Connector-attribute-file
  ```
  Example of execution
  ```
  cjgetresprop MyServer -type rar -resname account-rar -c AccountProp.xml
  ```
- Setting the attributes
  Execute the following command to apply the values of the Connector attribute file:
  Execution format
  ```
  cjsetresprop server-name [-nameserver provider-URL] -type rar -resname
  DB-Connector-display-name -c Path-of-Connector-attribute- file
  ```
  Example of execution
  ```
  cjsetresprop MyServer -type rar -resname account-rar -c AccountProp.xml
  ```

  Important note

  When deploying a resource adapter and then defining the properties, use the `cjgetrarprop` and `cjsetrarprop` commands. For details on the `cjgetrarprop` command, see *cjgetrarprop (get RAR file Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetrarprop` command, see *cjsetrarprop (HITACHI Connector Property Settings)* in the *uCosminexus Application Server Command Reference Guide*.

(3) Attribute settings to be edited

The settings for DB Connector properties are described below:

- General information of the DB Connector
- Configuration properties
- Runtime properties
- Optional name information

(a) General information of the DB Connector

The following table describes the settings for the general information attributes (`<outbound-resourceadapter>` tag) of the DB Connector that can be set:

Items	Mandatory	Corresponding tags
Transaction support level	Y	<transaction-support>
Scope of re-authentication support	Y	<reauthentication-support>

Legend:
  Y: Mandatory

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Configuration properties

The following table describes the settings for the configuration properties (`<outbound-resourceadapter>` - `<connection-definition>` - `<config-property>` tag) of the DB Connector:

Items	Corresponding tags
Configuration property name	<config-property-name>
Configuration property data type	<config-property-type>
Configuration property value	<config-property-value>[#]

#

To clear the configuration property value, specify an empty tag (`<config-property-value></config-property-value>`).

If the `<config-property-value>` tag itself is not specified, the configuration property value is not changed.

Repeat the settings under the `<config-property>` tag for all the configuration properties that you want to define.

Some of the items that you need to set up differ based on the type of imported DB Connector. For details on the properties that can be set in the `<config-property>` tag, see *4.1.10 Properties that you can specify in the <config-property> tag set up for DB Connector* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

Note that when `XAOpenString` is specified in the `<config-property-name>` tag and if you execute the `cjgetresprop` command, the configuration property value (`<config-property-value>`) will be displayed as follows:

- When the property value is set up

  The configuration property value (`<config-property-value>`) is not obtained and *<!-- The config-property-value is already set up. -->* is displayed.

To change the value of `XAOpenString`, add the `<config-property-value>` tag, and then specify the changed value.

- When the property value is not set up

  An empty tag (`<config-property-value></config-property-value>`) is displayed.

  To set up the value of `XAOpenString`, add the value in the `<config-property-value>` tag.

For statement pooling operations and notes on using the statement pooling functionality, see *3.14.4 Statement pooling* in the *uCosminexus Application Server Common Container Functionality Guide*.

For details on the examples of configuration property settings for the DB Connector to be used, see *(4) Example of configuration property settings*.

(c) Runtime properties

The following table describes the settings for the runtime properties (`<outbound-resourceadapter>` - `<connection-definition>` - `<connector-runtime>` tag) of the DB Connector:

Items	Corresponding tags
Property name	<property-name>
Property data type	<property-type>
Property value	<property-value>[#]

#

To clear a property value, specify an empty tag (`<property-value></property-value>`).

If the `<property-value>` tag itself is not specified, the property value is not changed.

Repeat the above settings for all the properties that you want to define.

When you set the property value (<property-value>), the value set as the default property value (<property-default-value>) will not be applied.

Set the following items in the property names (<property-name>):

Property items	Item names of property name (<property-name>)
User name[#]	User
Password[#]	Password
Minimum number of connections to be pooled in the connection pool	MinPoolSize
Maximum number of connections to be pooled in the connection pool	MaxPoolSize
Choose a method to check whether a failure has occurred in the connections inside the pool	ValidationType
Interval for periodically checking whether a failure has occurred in the connections inside the pool	ValidationInterval
Retry count when you fail to acquire the connection	RetryCount
Retry interval when you fail to acquire the connection	RetryInterval
Choose whether to output the log	LogEnabled

Property items	Item names of property name (<property-name>)
Period from the last usage of connection until you determine whether to cancel the connection automatically (connection sweeper)	ConnectionTimeout
Interval for canceling the connection automatically (connection sweeper)	SweeperInterval
Choose whether to manage the requests for connection acquisition in a queue when the connections are used up	RequestQueueEnable
Maximum waiting time when the requests for connection acquisition (when the connections are used up) are managed in a queue	RequestQueueTimeout
Choose whether to enable alert output for monitoring the connection pool	WatchEnabled
Interval for monitoring the connection pool	WatchInterval
Threshold value for monitoring the usage state of the connection pool	WatchThreshold
Choose whether to output the connection pool monitoring results to a file	WatchWriteFileEnabled
Interval for operating the connection count adjustment function	ConnectionPoolAdjustmentInterval
Choose whether to enable the warming up function of the connection pool	Warmup
Choose whether to enable the timeout of the network failure detection function	NetworkFailureTimeout

#

When the `User`, `Password` is set in the <property-name> tag, and if you execute the cjgetresprop command, the property-value (<property-value>) will be displayed as described below:

- When the property value is set

  The property value (`<property-value>`) is not obtained and *<!-- The property-value is already set up. -->* is displayed.

  To change the value of *User* and *Password*, add the `<property-value>` tag, and then specify the changed value.

- When the property-value is not set

  An empty tag (`<property-value></property-value>`) is displayed.

  To set up the value of *User* and *Password*, add the value in the `<property-value>` tag.

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For connection pooling operations and notes on using the connection pooling functionality, see *3.14.1 Connection pooling* in the *uCosminexus Application Server Common Container Functionality Guide*.

(d) Optional name information

The following table describes the settings for the optional name information (`<outbound-resourceadapter>` - `<connection-definition>` - `<connector-runtime>` - `<resource-external-property>` tag) of the DB connector:

Items	Mandatory	Corresponding tags
Resource optional name	Y	<optional-name>
Resource authentication method	O	<res-auth>
Resource sharing scope	O	<res-sharing-scope>

Legend:
Y: Mandatory
O: Optional

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For details on how to use an optional name of the DB Connector, see *4.7 Referencing and changing the resource adapter names registered in the JNDI name space*.

(4) Example of configuration property settings

This subsection shows examples of configuration property settings for the following DB Connectors:

- When using HiRDB or XDM/RD E2 with `DBConnector_HiRDB_Type4_CP.rar`
- When using HiRDB with `DBConnector_HiRDB_Type4_XA.rar`
- When using Oracle with `DBConnector_Oracle_CP.rar`
- When using Oracle with `DBConnector_Oracle_XA.rar`
- When using SQL Server with `DBConnector_SQLServer_CP.rar`

(a) When using HiRDB or XDM/RD E2 with DBConnector_HiRDB_Type4_CP.rar

The following table describes an example of configuration property settings when using HiRDB or XDM/RD E2 with DBConnector_HiRDB_Type4_CP.rar:

Table 4–4: Example of configuration property settings when using HiRDB or XDM/RD E2 as a database (With DBConnector_HiRDB_Type4_CP.rar)

Item names	Example of settings for HiRDB	Example of settings for XDM/RD E2
Description	*HiRDB- port-number*#	*Server-schedule-number-of-the-database-connection-server*#
DBHostName	*HiRDB-host-name*	*XDM/RD-E2-host-name*
environmentVariables	*HiRDB-client-environment-variable-names*	*HiRDB-client-environment-variable-names*
loginTimeout	8	8
encodeLang	--	--
JDBC_IF_TRC	false	false
TRC_NO	500	500
uapName	--	--
LONGVARBINARY_Access	REAL	REAL
SQLInNum	300	300
SQLOutNum	300	300
SQLWarningLevel	SQLWARN	SQLWARN
SQLWarningIgnore	false	false
HiRDBCursorMode	false	false
maxBinarySize	0	0

Item names	Example of settings for HiRDB	Example of settings for XDM/RD E2
LONGVARBINARY_AccessSize	0	0
LONGVARBINARY_TruncError	true	true
PreparedStatementPoolSize	10	10
CallableStatementPoolSize	10	10
CancelStatement	true	true
logLevel	ERROR	ERROR

Legend:

-- : Settings are not required

#

You can also specify the environment variable group name (in Windows) or the path of environment variable group setup file (in UNIX) of HiRDB client.

(b) When using HiRDB with DBConnector_HiRDB_Type4_XA.rar

The following table describes an example of configuration property settings when using HiRDB with DBConnector_HiRDB_Type4_XA.rar:

Table 4–5: Example of configuration property settings when using HiRDB as database (With DBConnector_HiRDB_Type4_XA.rar)

Item names	Example of settings for HiRDB
Description	*Environment-variable-group-identifier*
DBHostName	*HiRDB-host-name*
environmentVariables	*HiRDB-client-environment-variable-names*
XAOpenString	*<Environment-variable-group-identifier*#1*> + <Path-of-environment-variable-group-setup-file*#2*>*
loginTimeout	8
encodeLang	--
JDBC_IF_TRC	false
TRC_NO	500
uapName	--
LONGVARBINARY_Access	REAL
SQLInNum	300
SQLOutNum	300
SQLWarningLevel	SQLWARN
SQLWarningIgnore	false
HiRDBCursorMode	false
maxBinarySize	0
LONGVARBINARY_AccessSize	0
LONGVARBINARY_TruncError	true

Item names	Example of settings for HiRDB
XACloseString	--
XALocalCommitMode	true
PreparedStatementPoolSize	10
CallableStatementPoolSize	10
CancelStatement	true
logLevel	ERROR

Legend:

--: Settings are not required

#1

Specify the value entered in the **Description** field.

#2

Specify the path of the environment variable group setup file of HiRDB. For details, see *4.1.6 Setting the database connection environment (setting HiRDB)* in the *uCosminexus Application Server System Setup and Operation Guide*.

(c) When using Oracle with DBConnector_Oracle_CP.rar

The following table describes the example of configuration property settings when using Oracle with `DBConnector_Oracle_CP.rar`:

Table 4–6: Example of configuration property settings when using Oracle as the database (with DBConnector_Oracle_CP.rar)

Item names	If a URL is not used	If a URL is used
databaseName	*Oracle-SID*	--
serverName	*Oracle-host-name* or *IP-address*	--
portNumber	1521	--
url	--	jdbc:oracle:thin:@//*Oracle-Database-host-name-or-IP-address*:1521/*Oracle-SID*
loginTimeout	8000	
PreparedStatementPoolSize	10	
CallableStatementPoolSize	10	
CancelStatement	true	
ConnectionIDUpdate	false	
logLevel	ERROR	

Legend:

--: Settings are not required

(d) When using Oracle with DBConnector_Oracle_XA.rar

The following table describes the example of configuration property settings when using Oracle with `DBConnector_Oracle_XA.rar`:

Table 4–7: Example of configuration property settings when using Oracle as the database (with DBConnector_Oracle_XA.rar)

Item names	If a URL is not used	If a URL is used
databaseName	*Oracle-SID*	--
serverName	*Oracle-host-name* or *IP-address*	--
portNumber	1521	--
url	--	jdbc:oracle:thin:@//*Oracle-Database-host-name-or-IP-address*:1521/*Oracle-SID*
loginTimeout	8000	
sessionTimeout	300	
PreparedStatementPoolSize	10	
CallableStatementPoolSize	10	
CancelStatement	true	
ConnectionIDUpdate	false	
logLevel	ERROR	

Legend:

--: Settings are not required

(e) When using SQL Server with DBConnector_SQLServer_CP.rar

The following table describes the example of configuration property settings when using SQL Server with `DBConnector_SQLServer_CP.rar`.

Table 4–8: When using SQL Server as the database

Item names	Example of settings for SQL Server
databaseName	*SQL-Server-database-name*
serverName	*SQL-Server-host-name-or-IP address*
applicationName	*Name-of-application-to-be-connected-to- SQL-Server*
instanceName	*Name-of-instance-to-be-connected-to- SQL-Server*
lastUpdateCount	true
lockTimeout	-1
loginTimeout	8
portNumber	1433
selectMethod	cursor
sendStringParametersAsUnicode	true
workstationID	*Host-name-of-Application Server*
xopenStates	false
failoverPartner	*Name-of-failover-server-used-in-database-mirroring-configuration*
integratedSecurity	false

Item names	Example of settings for SQL Server
packetSize	4096
PreparedStatementPoolSize	10
CallableStatementPoolSize	10
CancelStatement	true
logLevel	ERROR
applicationIntent	ReadWrite
multiSubnetFailover	false

4.2.3 Deploying the DB Connector

If you deploy a DB Connector, you can use it as a J2EE resource adapter. The *J2EE resource adapter* is a resource adapter that is deployed as a shared standalone module on the J2EE server. If you deploy a resource adapter imported by using the server management commands, that resource adapter becomes available to all the J2EE applications running on that J2EE server. Note that you can also define the properties after deploying the DB Connector. In such a case, however, define the properties when the applicable DB Connector is not running. For details on how to define the properties, see *4.2.2 Defining the DB Connector properties*.

Execute the following command to deploy the DB Connector:

Execution format

```
cjdeployrar [server-name] [-nameserver provider-URL] -resname DB-Connector
-display-name
```

Example of execution

```
cjdeployrar MyServer -resname account-rar
```

For details on the `cjdeployrar` command, see *cjdeployrar (deploy resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

4.2.4 Testing the connectivity of DB Connector

Use the connectivity test to verify whether the information set in the DB connector is correct.

Execute the following command to test the connectivity of DB Connector:

Execution format

```
cjtestres [server-name] [-nameserver provider-URL] -type rar -resname DB-C
onnector-display-name
```

Example of execution

```
cjtestres MyServer -type rar -resname account-rar
```

For details on the `cjtestres` command, see *cjtestres (execute resource connection test)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

You cannot delete the DB Connector for which connection has been tested once until you restart the J2EE server. To delete the DB Connector, stop the DB Connector and then restart the J2EE server.

4.2.5 Starting the DB Connector

Execute the following command to start the DB Connector:

Execution format

```
cjstartrar [server-name] [-nameserver provider-URL] -resname DB-Connector
-display-name
```

Example of execution

```
cjstartrar MyServer -resname account-rar
```

For details on the `cjstartrar` command, see *cjstartrar (start resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- If a J2EE resource in the J2EE application references the DB Connector, start the DB Connector and then start the J2EE application.
- You cannot delete a DB Connector that has been started until you restart the J2EE server. To delete the DB Connector, stop the DB Connector and then restart the J2EE server.

4.2.6 Stopping the DB Connector

Execute the following command to stop the DB Connector:

Execution format

```
cjstoprar [server-name] [-nameserver provider-URL] -resname DB-Connector-d
isplay name
```

Example of execution

```
cjstoprar MyServer -resname account-rar
```

For details on the `cjstoprar` command, see *cjstoprar (stop resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

If a J2EE resource in the J2EE application references the DB Connector, stop the J2EE application and then stop the DB Connector.

4.2.7 Undeploying the DB Connector

Preparation

Before you delete a deployed DB Connector, stop the DB Connector and restart the J2EE server. Likewise, if you try to start or test the connectivity of the DB Connector even once, restart the J2EE server.

Execute the following command to un-deploy the DB Connector:

Execution format

```
cjundeployrar [server-name] [-nameserver provider-URL] -resname DB-Connect
or-display-name
```

Example of execution

```
cjundeployrar MyServer -resname account-rar
```

For details on the `cjundeployrar` command, see *cjundeployrar (undeploy resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

4.2.8 Exporting the DB Connector

You output (export) the contents of the DB Connector as a RAR file.

Execute the following command to export the DB Connector:

Execution format

```
cjexportrar [server-name] [-nameserver provider-URL] -f file-path -resnam
e DB-Connector-display-name
```

Example of execution

```
cjexportrar MyServer -f res1.rar -resname account-rar
```

For details on the `cjexportrar` command, see *cjexportrar (export resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

When you operate with the Management Server and if the server management commands and the host of the Management Server are different, you need to export the DB Connector and store it in the host of Management Server.

4.3 Settings for connecting to other resources

Use the resource adapter for connecting to resources like OpenTP1. This section explains the basic settings for connecting to the resource adapters other than DB Connector, DB Connector for Cosminexus RM, and Cosminexus RM.

Use the following procedure for setting a resource adapter:

1. Import the resource adapter.
2. Define the properties.
3. Deploy the resource adapter.
   Deploying the resource adapter refers to the process of deploying the resource adapter as a shared and standalone module (J2EE resource adapter) on the J2EE server.
4. Check the connectivity.
   You can verify if the connection is correct using a connectivity test.

For a deployed J2EE resource adapter, you need to resolve the resource adapter references by using the J2EE application property settings. For details, see *9.3.3 Defining resource adapter references*.

Tip

You can use the following resource adapters to connect with other resources:

- Resource adapters compliant with Connector 1.0
- Resource adapters compliant with Connector 1.5

Note that the resource adapters provided with Application Server are compliant with Connector 1.0 specifications.

For details on defining the properties of resource adapters compliant with Connector 1.0 specifications, see *4.3.2 Defining the resource adapter properties (for Connector 1.0)* and for defining the properties of resource adapters compliant with Connector 1.5 specifications, see *4.3.3 Defining the resource adapter properties (for Connector 1.5)*.

4.3.1 Importing the resource adapter

You import the resource adapter.

Note that when importing independent resource adapters other than those provided by Application Server, import the resource adapters as per the description given in *Notes*.

Execute the following command to import the resource adapters:

Execution format

```
cjimportres [server-name] [-nameserver provider-URL] -type rar -f file-pa
th
```

Example of execution

```
cjimportres MyServer -type rar -f mqcadpt.rar
```

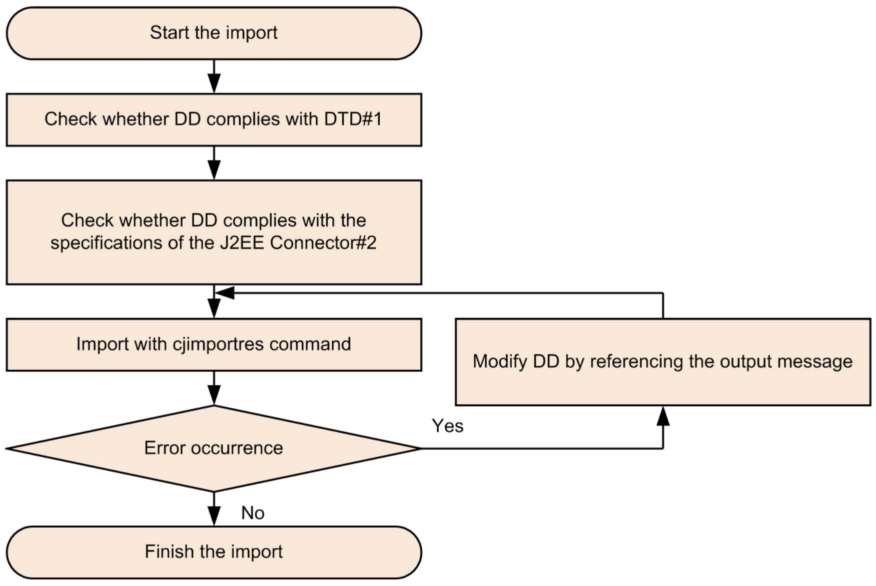
For details on the `cjimportres` command, see *cjimportres (import resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- You cannot import the RAR file with a DD not conforming to the specifications of J2EE Connector 1.0. You can only import a RAR file with a DD conforming to the specifications of J2EE Connector 1.0. Check whether the DD is as per the DTD and then import the resource adapter.

  The following figure shows the procedure for importing the resource adapters:

  Figure 4–1: Procedure for importing the resource adapter (RAR)

  
  Start the import
Check whether DD complies with DTD#1
Check whether DD complies with the specifications of the J2EE Connector#2
Import with cjimportres command
Modify DD by referencing the output message
Yes
Error occurrence
No
Finish the import

  #1

  The DD is handled as a validated XML document. Therefore, an error occurs either when the DOCTYPE declaration in the DD is wrong, or when the DD does not comply with the DTD specifications published by Sun Microsystems Inc.

  #2

  An error occurs when a value based on the J2EE Connector specifications is not specified in the DD. For details, see *4.1.26 Setting DB Connector (when using CUI)* in the *uCosminexus Application Server System Setup and Operation Guide*.

  If a message is output, see the manual *uCosminexus Application Server Messages* and take measures.

- The RAR file name specified when importing the resource adapter is used as the directory name in the work directory. The JAR file or the native library in the RAR file is deployed in the work directory. Specify the RAR file name and the file names in the RAR file so that the path length of the work directory does not reach the limit specified for the platform. For estimating the path length of the work directory for a system that executes J2EE applications, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*. For estimating the path length of the work directory for a system that executes batch applications, see *C.2 Work directory of the batch server* in the *uCosminexus Application Server System Setup and Operation Guide*.

- When the `<display-name>` tag of `ra.xml` is not set, the Display name is decided on the basis of the RAR file name.

- Do not include files with the following names in the RAR file:

```
rar.properties
fileinfo.properties
META-INF/hitachi-ra.xml
```

4.3.2 Defining the resource adapter properties (for Connector 1.0)

Define the properties of resource adapters compliant with Connector 1.0 specifications. You can also define the properties of the resource adapter after deploying the J2EE resource adapter. Note that if you want to change the properties of a resource adapter that is already set, change the properties when the applicable resource adapter is not running.

For an overview of the property setup procedures, see *3.5 Property settings using the property file*. The following subsections define the resource adapter properties:

(1) Property file to be edited

HITACHI Connector Property file

(2) Acquiring the property file to be edited and setting the properties

- Acquiring the property file
  Execute the following command to acquire the HITACHI Connector Property file of the resource adapter:

  Execution format

  ```
  cjgetresprop [server-name] [-nameserver provider-URL] -type rar -resnam
  e display-name-of-the-resource-adapter -c path-of-Connector-attribute-f
  ile
  ```

  Example of execution

  ```
  cjgetresprop MyServer -type rar -resname Rar1 -c AccountProp.xml
  ```

- Setting the attributes
  Execute the following command to apply the values of the Connector attribute file:

  Execution format

  ```
  cjsetresprop [server-name] [-nameserver provider-URL] -type rar -resnam
  e display-name-of-the-resource-adapter -c path-of-Connector-attribute-f
  ile
  ```

  Example of execution

  ```
  cjsetresprop MyServer -type rar -resname Rar1 -c AccountProp.xml
  ```

  Important note

  When defining the properties after deploying the resource adapter, use the `cjgetrarprop` command and `cjsetrarprop` command. For details on the `cjgetrarprop` command, see *cjgetrarprop (get RAR file Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetrarprop` command, see *cjsetrarprop (HITACHI Connector Property Settings)* in the *uCosminexus Application Server Command Reference Guide*.

(3) Property settings to be edited

The property settings for the resource adapter are described below:

- General information of the resource adapter
- Configuration properties
- Runtime properties
- Optional name information

(a) General information of the resource adapter

The following table describes the settings for general information (`<outbound-resourceadapter>` tag) of the resource adapter:

Items	Mandatory	Corresponding tags
Transaction support level	Y	<transaction-support>
Scope of re-authentication support	Y	<reauthentication-support>

Legend:
 Y: Mandatory

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Configuration properties

The following table describes the settings for the configuration properties (`<outbound-resourceadapter>` - `<connection-definition>` - `<config-property>` tag) of the resource adapter:

Items	Corresponding tags
Configuration property name	<config-property-name>
Configuration property data type	<config-property-type>
Configuration property value	<config-property-value>

Repeat the above settings for all the configuration properties that you want to define.

For the configuration property name (`<config-property-name>`) to be defined, see *4.2.2 Defining the DB Connector properties*.

(c) Runtime properties

The following table describes the settings for runtime properties (<property> tag) of the resource adapter:

Items	Corresponding tags
Property name	<property-name>
Property data type	<property-type>
Property value	<property-value>

Repeat the above settings for all the properties that you want to define.

For details on the runtime property names (<property-name>) to be defined and operations and precautions for the connection pool, see *4.2.2 Defining the DB Connector properties*.

(d) Optional name information

The following table describes the settings for optional name information (`<outbound-resourceadapter>`-`<connection-definition>`-`<connector-runtime>`-`<resource-external-property>` tag) of the resource adapter:

Items	Mandatory	Corresponding tags
Optional name of the resource	Y	<optional-name>
Resource authentication method	O	<res-auth>
Scope of resource sharing	O	<res-sharing-scope>

Legend:
Y: Mandatory
O: Optional

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For details on how to use the optional name of the resource adapter, see *4.7 Referencing and changing the resource adapter names registered in the JNDI name space*.

4.3.3 Defining the resource adapter properties (for Connector 1.5)

Define the properties for connecting to the resource adapter compliant with Connector 1.5 specifications.

You can also define the properties of the resource adapter after deploying the J2EE resource adapter. Note that if you want to change the properties of a resource adapter that is already set, change the properties when the applicable resource adapter is not running.

For an overview of the property setup procedures, see *3.5 Property settings using the property file*. The following subsections define the resource adapter properties:

(1) Property file to be edited

HITACHI Connector Property file

(2) Acquiring the property file to be edited and setting the properties

- Acquiring the property file
  Execute the following command to acquire the Connector property file of the resource adapter:

  Execution format

```
cjgetresprop [server-name] [-nameserver provider-URL] -type rar -resnam
e display-name-of-the-resource-adapter -c path-of-Connector-attribute-f
ile
```

Example of execution

```
cjgetresprop MyServer -type rar -resname Rar1 -c AccountProp.xml
```

- Setting the properties

  Execute the following command to apply the values of the HITACHI Connector Property file:

  Execution format

  ```
  cjsetresprop [server-name] [-nameserver provider-URL] -type rar -resnam
  e display-name-of-the-resource-adapter -c path-of-Connector-attribute-f
  ile
  ```

  Example of execution

  ```
  cjsetresprop MyServer -type rar -resname Rar1 -c AccountProp.xml
  ```

  Important note

  When defining the properties after deploying the resource adapter, use the `cjgetrarprop` command and `cjsetrarprop` command. For details on the `cjgetrarprop` command, see *cjgetrarprop (get RAR file Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the `cjsetrarprop` command, see *cjsetrarprop (HITACHI Connector Property Settings)* in the *uCosminexus Application Server Command Reference Guide*.

(3) Property settings to be edited

A resource adapter compliant with Connector 1.5 specifications might include multiple connection definitions.

Specify the following property settings:

- **Property settings of resource adapters**

  Specify the following properties:

  - Configuration properties
  - Outbound resource adapter properties
  - Inbound resource adapter properties
  - Properties of the administered objects
  - Runtime properties

  For details on the property settings, see *(4) Setting the resource adapter properties*.

- **Property settings for each Outbound resource adapter connection**

  For using Outbound resource adapters, specify the following properties:

  - Configuration properties
  - Runtime properties
  - Optional name information

  For details on the property settings for each Outbound resource adapter connection, see *(5) Property settings for each Outbound resource adapter connection*.

(4) Setting the resource adapter properties

The following sections describe the settings for the resource adapter properties:

(a) Configuration properties

The following table describes the settings for the configuration properties (`<resourceadapter>` - `<config-property>` tag) of the resource adapter:

Items	Corresponding tags
Configuration property name	<config-property-name>
Configuration property data type	<config-property-type>
Configuration property value	<config-property-value>

Repeat the above settings for all the configuration properties that you want to define.

For the configuration property name (<config-property-name>) to be defined, see *4.2.2 Defining the DB Connector properties*.

(b) Outbound resource adapter properties

The following table describes the settings for the Outbound resource adapter properties (`<outbound-resourceadapter>` tag):

Items	Mandatory	Corresponding tags
Property settings for each connection#	Y	<connection-definition>
Transaction support level	O	<transaction-support>
Scope of re-authentication support	O	<reauthentication-support>

Legend:

Y: Mandatory

O: Optional

#

For details on the property settings for each Outbound resource adapter connection, see *(5) Property settings for each Outbound resource adapter connection*.

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(c) Inbound resource adapter properties

The following table describes the settings for the Inbound resource adapter properties (`<inbound-resourceadapter>` - `<messageadapter>` - `<messagelistener>` tag):

Items	Corresponding tags
Message listener type	<messagelistener-type>
Information about ActivationSpec	<activationspec>

(d) Properties of the administered objects

The following table describes the property settings for the administered objects (`<adminobject>` - `<config-property>` tag) of the resource adapter:

Items	Corresponding tags
Property name of the administered object	<config-property-name>
Property data type of the administered object	<config-property-type>
Property value of the administered object	<config-property-value>

(e) Runtime properties

The following table describes the settings for runtime properties (`<resourceadapter-runtime>` - `<property>` tag) of the resource adapter:

Items	Corresponding tags
Property name	<property-name>
Property data type	<property-type>
Property value	<property-value>[#]

#

For details on how to set up property values, see *4.2.2 Defining the DB Connector properties*.

Repeat the above settings for all the properties that you want to define.

Specify the following settings in the property name (`<property-name>`):

Property items	Item name of property name (<property-name>)
Maximum number of threads executed concurrently in the thread pool	MaxTPoolSize
Minimum number of threads in the thread pool	MinTPoolSize
Value of a timeout until a thread is released from the thread pool (seconds)	TPoolKeepalive

Note:

Specify the above settings when the lifecycle management function is enabled. When the lifecycle management function is not used (`<resourceadapter-class>` is not specified), the property value (`<property-value>`) is ignored. For the lifecycle management functionality, see *3.16.1 Lifecycle management of a resource adapter* in the *uCosminexus Application Server Common Container Functionality Guide*.

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(5) Property settings for each Outbound resource adapter connection

The property settings for each Outbound resource adapter connection are described as follows:

(a) Configuration properties

The following table describes the settings for the configuration properties (`<outbound-resourceadapter>` - `<connection-definition>` - `<config-property>` tag) of the Outbound resource adapter:

Items	Corresponding tags
Configuration property name	<config-property-name>
Configuration property data type	<config-property-type>
Configuration property value	<config-property-value>

Repeat the above settings for all the configuration properties that you want to define.

For the configuration property name (<config-property-name>) to be defined, see *4.2.2 Defining the DB Connector properties*.

Note that the configuration properties for each connection combine the configuration properties specified for each connection (`<outbound-resourceadapter>` - `<connection-definition>` - `<config-property>` tag) and the configuration properties specified for the resource adapter (`<resourceadapter>` - `<config-property>` tag). If the same configuration property name is specified, the priority will be given to the configuration property specified for each connection.

(b) Runtime properties

The following table describes the settings for runtime properties (`<outbound-resourceadapter>` - `<connection-definition>` - `<connector-runtime>` tag) of Outbound resource adapters:

Items	Corresponding tags
Property name	<property-name>
Property data type	<property-type>
Property value	<property-value>

Repeat the above settings for all the properties that you want to define.

For details on the runtime property names (<property-name>) to be defined and operations and precautions for the connection pool, see *4.2.2 Defining the DB Connector properties*.

(c) Optional name information

The following table describes the settings for optional name information (`<outbound-resourceadapter>` - `<connection-definition>` - `<connector-runtime>` - `<resource-external-property>` tag) of Outbound resource adapters:

Items	Mandatory	Corresponding tags
Optional name of the resource	Y	<optional-name>
Resource authentication method	O	<res-auth>
Scope of resource sharing	O	<res-sharing-scope>

Legend:
Y: Mandatory
O: Optional

For details on the property settings, see *4.1.1 Specifications of the HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For details on how to use the optional name of the resource adapter, see *4.7 Referencing and changing the resource adapter names registered in the JNDI name space*.

4.3.4 Deploying the resource adapter

If you deploy a resource adapter, you can use it as a J2EE resource adapter.

If you deploy a resource adapter imported by using the server management commands, that resource adapter becomes available to all the J2EE applications running on that J2EE server. Note that you can define the properties of the resource adapter after deploying the resource adapter. For details on how to define the properties, see *4.3.2 Defining the resource adapter properties (for Connector 1.0)* or *4.3.3 Defining the resource adapter properties (for Connector 1.5)* depending on the connector architecture specifications supported in resource adapters.

Execute the following command to deploy the resource adapter:

Execution format

```
cjdeployrar [server-name] [-nameserver provider-URL] -resname display-name
-of-the-resource-adapter [-resname display-name-of-the-resource-adapter]
```

Example of execution

```
cjdeployrar MyServer -resname Rar1
```

For details on the `cjdeployrar` command, see *cjdeployrar (deploy resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

If you specify a resource adapter containing the runtime information of a running resource adapter, the resource adapter is started automatically after the deployment is complete. This J2EE resource adapter is not removed even if the auto start process fails.

4.3.5 Testing the connectivity of a J2EE resource adapter

Use the connectivity test to verify whether the information set in the resource adapter is correct.

Execute the following command to test the connectivity of the resource adapter:

Execution format

```
cjtestres [server-name] [-nameserver provider-URL] -type rar -resname disp
lay-name-of-the-resource-adapter [-resname display-name-of-the-resource-ad
apter]
```

Example of execution

```
cjtestres MyServer -type rar -resname Rar1
```

For details on the `cjtestres` command, see *cjtestres (execute resource connection test)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- You cannot delete a resource adapter for which connectivity has been tested once until you restart the J2EE server. To delete the resource adapter, stop the resource adapter and then restart the J2EE server.
- If resource adapters compliant with Connector 1.5 specifications have multiple connection definitions, the connectivity test is executed for all the connection definitions. Even if an error occurs in one of the connection definitions, the connectivity test will be executed for all the connection definitions. However, in such a case, the result of the connectivity test will be "Failed".
- If resource adapters are compliant with Connector 1.5 specifications, a connectivity test can be executed only when Outbound resource adapters are included.

4.3.6 Starting the J2EE resource adapter

You start the J2EE resource adapter.

Execute the following command to start the J2EE resource adapter:

Execution format

```
cjstartrar [server-name] [-nameserver provider-URL] -resname display-name
-of-the-resource-adapter
```

Example of execution

```
cjstartrar MyServer -resname Rar1
```

For details on the `cjstartrar` command, see *cjstartrar (start resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- If a J2EE resource in the J2EE application references a J2EE resource adapter, start the J2EE resource adapter and then start the J2EE application.
- You cannot delete a J2EE resource adapter once started until you restart the J2EE server. To delete the J2EE resource adapter, stop the J2EE resource adapter, restart the J2EE server, and confirm that the J2EE resource adapter you are trying to delete is outside the scope of reference resolution by a J2EE application.

4.3.7 Stopping the J2EE resource adapter

You stop the J2EE resource adapter.

Execute the following command to stop the J2EE resource adapter:

Execution format

```
cjstoprar [server-name] [-nameserver provider-URL] -resname display-name-o
f-the-resource-adapter
```

Example of execution

```
cjstoprar MyServer -resname Rar1
```

For details on the `cjstoprar` command, see *cjstoprar (stop resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

If a J2EE resource in the J2EE application references the J2EE resource adapter, stop the J2EE application and then stop the J2EE resource adapter.

4.3.8 Undeploying the J2EE resource adapter

You delete the deployed J2EE resource adapter.

Preparation

Before you delete a deployed J2EE resource adapter, stop the J2EE resource adapter and restart the J2EE server. Likewise, if you try to start or test the connectivity of the J2EE resource adapter even once, restart the J2EE server.

Execute the following command to un-deploy the J2EE resources:

Execution format

```
cjundeployrar [server-name] [-nameserver provider-URL] -resname display-na
me-of-the-resource-adapter
```

Example of execution

```
cjundeployrar MyServer -resname Rar1
```

For details on the `cjundeployrar` command, see *cjundeployrar (undeploy resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

4.3.9 Exporting the J2EE resource adapter

You output (export) the contents of the J2EE resource adapter as a RAR file.

Execute the following command to export the J2EE resource adapter:

Execution format

```
cjexportrar [server-name] [-nameserver provider-URL] -f file-path -resnam
e display-name-of-J2EE-resource-adapter
```

Example of execution

```
cjexportrar MyServer -f res1.rar -resname Rar1
```

For details on the `cjexportrar` command, see *cjexportrar (export resource adapter)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

When you operate with Management Server and if the host for executing the server management commands and the host of the Management Server are different, you need to export the J2EE resource adapter and store it in the host of Management Server.

4.4 Displaying the state and list of J2EE resource adapters

You can display the adapter names and states of the deployed resource adapters. For the resource adapters compliant with Connector 1.5 specifications, you can also display the information such as the list of connection definition identifiers and list of message listener types.

4.4.1 Displaying the J2EE resource adapter state

Execute the following command to display the names and states (running or terminated) of all the deployed resource adapters:

Execution format

```
cjlistrar [server-name] [-nameserver provider-URL] [-spec]
```

Example of execution

```
cjlistrar MyServer
```

If you specify the `-spec` option, the version of the connector architecture is displayed in the list of states for resource adapters.

For details on the `cjlistrar` command, see *cjlistrar (list resource adapters)* in the *uCosminexus Application Server Command Reference Guide*.

4.4.2 Displaying the list of connection definition identifiers (outbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display the list of connection definition identifiers of the deployed Outbound resource adapters:

Execution format

```
cjlistrar [server-name] [-nameserver provider-URL] -resname RAR-display-na
me -outbound
```

Example of execution

```
cjlistrar MyServer -resname account-rar -outbound
```

For details on the `cjlistrar` command, see *cjlistrar (list resource adapters)* in the *uCosminexus Application Server Command Reference Guide.*

4.4.3 Displaying the list of message listener types (Inbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display the list of message listener types of the deployed Inbound resource adapters:

Execution format

```
cjlistrar [server-name] [-nameserver provider-URL] -resname RAR-display-na
me -inbound
```

Example of execution

```
cjlistrar MyServer -resname account-rar -inbound
```

For details on the `cjlistrar` command, see *cjlistrar (list resource adapters)* in the *uCosminexus Application Server Command Reference Guide*.

4.4.4 Displaying the list of property names required for activating the message listeners (Inbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display the list of property names required for activating the message listeners of the deployed Inbound resource adapters:

Execution format

```
cjlistrar [server-name] [-nameserver provider-URL] -resname RAR-display-na
me -listenertype name-of-the-message-listener-type
```

Example of execution

```
cjlistrar MyServer -resname account-rar -listenertype javax.jms.MessageLis
tener
```

For details on the `cjlistrar` command, see *cjlistrar (list resource adapters)* in the *uCosminexus Application Server Command Reference Guide*.

4.5 Displaying the list of resource adapters

You can display the list of imported resource adapters. For the resource adapters compliant with Connector 1.5 specifications, you can also display the information such as the list of connection definition identifiers and the list of message listener types.

4.5.1 Displaying the list of resource adapters

Execute the following command to display the list of imported resource adapters:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type rar [-spec]
```

Example of execution

```
cjlistres MyServer -type rar
```

If you specify the `-spec` option, the version of the connector architecture is displayed in the list.

For details on the `cjlistres` command, see *cjlistres (list resources)* in the *uCosminexus Application Server Command Reference Guide*.

4.5.2 Displaying the list of connection definition identifiers (Outbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display the list of connection definition identifiers of the Outbound resource adapters deployed as J2EE resource adapters:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type rar -resname RAR
-display-name -outbound
```

Example of execution

```
cjlistres MyServer -type rar -resname account-rar -outbound
```

For details on the `cjlistres` command, see *cjlistres (list resources)* in the *uCosminexus Application Server Command Reference Guide*.

4.5.3 Displaying the list of message listener types (Inbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display a list of message listener types of the Inbound resource adapters deployed as J2EE resource adapters:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type rar -resname RAR
-display-name -inbound
```

Example of execution

```
cjlistres MyServer -type rar -resname account-rar -inbound
```

For details on the `cjlistres` command, see *cjlistres (list resources)* in the *uCosminexus Application Server Command Reference Guide*.

4.5.4 Displaying the list of property names required for activating the message listeners (Inbound resource adapter)

Execute the following command for the resource adapters compliant with Connector 1.5 specifications to display the list of property names required for activating the message listeners of the Inbound resource adapters deployed as J2EE resource adapters:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type rar -resname RAR
-display-name -listenertype name-of-the-message-listener-type
```

Example of execution

```
cjlistres MyServer -type rar -resname account-rar -listenertype javax.jms.
MessageListener
```

For details on the `cjlistres` command, see *cjlistres (list resources)* in the *uCosminexus Application Server Command Reference Guide*.

4.6 Checking the connection pool state

Display the information and connection state (used or unused) of the resource adapter connection pool. Moreover, delete resource adapter connections as and when required.

4.6.1 Displaying the connection pool state

You use the following command to display the information and state of resource adapter connection pools:

Execution format

```
cjlistpool [server-name] [-nameserver provider-URL] -resname display-name
-of-the-resource-adapter
```

Example of execution

```
cjlistpool MyServer -resname Rar1
```

For details on the `cjlistpool` command, see *cjlistpool (list connection pools)* in the *uCosminexus Application Server Command Reference Guide*.

Reference note

For the resource adapters compliant with Connector 1.5 specifications, specify the *display-name-of-the-resource-adapter* of the `-resname` option in the following format:

display-name-of-the-resource-adapter!*connection-definition-identifier*

4.6.2 Deleting the connection pools

You use the following command to delete the connection pools of resource adapters. This command will delete all the unused connection pools.

Execution format

```
cjclearpool [server-name] [-nameserver provider-URL] -type connector -resn
ame display-name-of-the-resource-adapter
```

Example of execution

```
cjclearpool MyServer -type connector -resname Rar1
```

For details on the `cjclearpool` command, see *cjclearpool (delete connection in connection pool)* in the *uCosminexus Application Server Command Reference Guide*.

Reference note

For the resource adapters compliant with Connector 1.5 specifications, specify the *display-name-of-the-resource-adapter* for the `-resname` option in the following format:

display-name-of-the-resource-adapter!*connection-definition-identifier*

4.7 Referencing and changing the resource adapter names registered in the JNDI name space

Change the resource adapter names registered in the JNDI name space.

An optional name is also added in the resource adapter names. You can add optional names to reference the resource adapters from the JNDI name space using optional names. This functionality is called the *user-specified namespace functionality*.

For the JNDI name space of resource adapters, see *2.3 Binding and lookup of objects to the JNDI name space* in the *uCosminexus Application Server Common Container Functionality Guide*. For the user-specified name space functionality, see *2.6 Adding optional names to Enterprise Beans or J2EE resources (user-specified name space functionality)* in the *uCosminexus Application Server Common Container Functionality Guide*.

You can add optional names in the `<resource-external-property>` tag of the HITACHI Connector Property file. However, you cannot add an optional name to a resource adapter that uses the JMS interface.

For details on editing the `<resource-external-property>` tag of the HITACHI Connector Property file, see the settings for the optional name information in *4.2.2 Defining the DB Connector properties*, *4.3.2 Defining the resource adapter properties (for Connector 1.0)*, or *4.3.3 Defining the resource adapter properties (for Connector 1.5)* depending on types of the used resource adapters.

Note:

If a running J2EE application is present, you cannot stop and delete the J2EE resource with the optional name specified. Stop all the J2EE applications running on the J2EE server.

4.8 Deleting the resource adapter

Execute the following command to delete the resource adapter:

Execution format

```
cjdeleteres [server-name] [-nameserver provider-URL] -type rar -resname di
splay-name-of-the-resource-adapter
```

Example of execution

```
cjdeleteres MyServer -type rar -resname account-rar
```

For details on the `cjdeleteres` command, see *cjdeleteres (delete resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

When the J2EE resource is deleted from the applications that contain `cosminexus.xml`, you cannot delete the definition information unique to the Application Server-related J2EE resources to be deleted that are included in `cosminexus.xml`. For details on deleting J2EE resources from the applications that contain `cosminexus.xml`, see *13.3.6 Running an application that includes cosminexus.xml* in the *uCosminexus Application Server Common Container Functionality Guide*.

4.9 Copying the resource adapter

Execute the following command to copy the resource adapter properties:

Execution format

```
cjcopyres [server-name] [-nameserver provider-URL] -type rar -src display
-name-of-resource-adapter-of-copy-source -dst display-name-of-resource-ada
pter-of-copy-destination
```

Example of execution

```
cjcopyres MyServer -type rar -src account-rar -dst account-rar2
```

For details on the `cjcopyres` command, see *cjcopyres (copy resource)* in the *uCosminexus Application Server Command Reference Guide*.

5 Setting up Resource Adapters Included in J2EE Applications

This chapter describes the setup of the resource adapters included in J2EE applications.

5.1 Overview of settings of the resource adapters included in J2EE applications

The settings of resource adapters included in J2EE applications are the operations for enabling the usage of resource adapters in J2EE applications.

This section describes the types of resource adapters that you can include and use in J2EE applications, and gives an overview of the setup.

5.1.1 Available resource adapters

The following resource adapters can be included in J2EE applications:

- DB Connector
- uCosminexus TP1 Connector
- Other resource adapters compliant with Connector 1.0 or resource adapters compliant with Connector 1.5#

#

For the types of other resource adapters compliant with Connector 1.0 or the resource adapters compliant with Connector 1.5, see *3.3.2 Types of resource adapters* in the *uCosminexus Application Server Common Container Functionality Guide*.

Important note

You can use J2EE resource adapters and resource adapters included in a J2EE application concurrently from the J2EE application. However, you cannot use J2EE resource adapters and resource adapters included in a J2EE application with the same display name concurrently. If you specify the same display name, an error will occur.

5.1.2 Overview of settings and operations

The following table gives an overview of settings of the resource adapters included in J2EE applications.

Table 5–1: Overview of settings of the resource adapters included in J2EE applications

Settings		Contents	Reference
Create J2EE applications including resource adapters#	Adding resource adapters to J2EE applications	Import RAR files into J2EE servers and add to the J2EE applications.	*5.2*
	Importing J2EE applications including resource adapters	Import J2EE applications created in the application development environment and include RAR files into J2EE servers.	*5.3*
Defining resource adapter properties		Set up the following information for connecting to a database: • General information of resource adapters • Configuration properties • Runtime properties • Optional name information	*5.4*

Settings	Contents	Reference
Testing the connectivity of resource adapters	Verify that the contents set up in the resource adapters are correct.	*5.5*
Starting and stopping J2EE applications including resource adapters	Start or stop J2EE applications including resource adapters.	*5.6*
Referencing the list of resource adapters included in J2EE applications	Reference the list of resource adapters included in J2EE applications and the list of connection definition identifiers.	*5.7*
Displaying and deleting the list of connection pools	Reference the connection pool state of resource adapters included in J2EE applications. Furthermore, delete the connections as and when required.	*5.8*

# You use one of the following methods to create J2EE applications including resource adapters:

- Add resource adapters to J2EE applications.
  Add RAR files to J2EE applications using the same method that is used for adding the EJB-JAR or WAR files to J2EE applications.
- Import J2EE applications including resource adapters.
  Create a J2EE application including resource adapters in the application development environment and import that J2EE application into a J2EE server.

5.2 Adding resource adapters to J2EE applications

You add resource adapters to J2EE applications.

Before adding a resource adapter, confirm that another resource adapter is not deployed with the same display name or the optional name as the resource adapter you want to add. If another resource adapter with the same display name or the optional name is deployed, you cannot add the resource adapter to the J2EE application. Note that you can use the same display name or the optional name as that of the resource adapters included in other J2EE applications.

To add resource adapters to J2EE applications:

1. Import RAR files of resource adapters into a J2EE server.
   For details on importing RAR files to be added to J2EE applications, see *7.2.3 Importing resource adapters (RAR files)*.

2. Add the imported RAR files to J2EE applications.
   For details on adding the resource adapters (RAR files) to J2EE applications, see *7.2.4(3) Adding RAR files*.

5.3 Importing J2EE applications including resource adapters

You import J2EE applications including resource adapters.

Before importing J2EE applications, confirm that a resource adapter is not deployed with the same display name or the optional name as that of the resource adapter that is included in the J2EE application. If another resource adapter with the same display name or the optional name is deployed, you cannot import that J2EE application including the resource adapter. Note that you can use the same display name or the optional name as that of the resource adapters that are included in other J2EE applications.

Import the following J2EE applications:

- J2EE applications in the archive format

  The J2EE applications in the archive format contain the components of EJB-JAR, WAR, or RAR files under the working directory of a J2EE server. For details on importing J2EE applications in the archive format, see *8.1.1 Importing the J2EE applications that are in archive format*.
- J2EE applications in the expanded archive format

  The J2EE applications in the expanded archive format are the J2EE applications containing the application entities such as EJB-JAR or WAR files in the files and directories that are external to the J2EE server, and follow fixed rules. However, the RAR files included in J2EE applications in the expanded archive format are saved in the archive format and not in the expanded file format. For details on importing J2EE applications in the expanded archive format, see *8.1.2 Importing the J2EE applications that are in exploded archive format*.

5.4 Defining resource adapter properties

You specify the settings of the resource adapters included in J2EE applications.

Set up the resource adapter properties when the J2EE application including resource adapters is not running.

For details on the property settings, see *3.5 Property settings using the property file*.

5.4.1 Property files to be edited

You can set up the properties of the resource adapters included in J2EE applications using one of the following property files:

- Use the HITACHI Connector Property file.
- Use the HITACHI Application Integrated Property File.
  Among the property files with components configuring the J2EE application integrated property file, set up the properties using the HITACHI Connector Property elements.

With the property settings of the components configuring J2EE applications, you can use the HITACHI Connector Property file and the HITACHI Application Integrated Property File individually and also in combination.

This section describes the property settings using the HITACHI Connector Property file. For details on the procedure of property settings using the HITACHI Application Integrated Property File, see *9.2 Property settings using the HITACHI Application Integrated Property File*.

5.4.2 Acquiring property files to be edited and setting up the attributes

- Acquiring property files
  Execute the following command to acquire a resource adapter property file:
  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type rar -resname display-name-of-resource-adapter -c path
  -of-the-HITACHI-Connector-Property-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name App1 -type rar -resname account-rar -c Acco
  untProp.xml
  ```

- Setting up the attributes
  Execute the following command to apply the values of resource adapter property files:
  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type rar -resname display-name-of-resource-adapter -c path
  -of-the-HITACHI-Connector-Property-file
  ```

Example of execution

```
cjsetappprop MyServer -name App1 -type rar -resname account-rar -c Acco
untProp.xml
```

5.4.3 Attribute settings to be edited

Define the properties according to the resource adapter type.

- For resource adapters compliant with Connector 1.0 specifications
  For details on the property settings of the resource adapters compliant with Connector 1.0 specifications, see *(3) Property settings to be edited* in *4.3.2 Defining the resource adapter properties (for Connector 1.0)*.
- For resource adapters compliant with Connector 1.5 specifications
  For details on the property settings of the resource adapters compliant with Connector 1.5 specifications, see *(3) Property settings to be edited* in *4.3.3 Defining the resource adapter properties (for Connector 1.5)*.

5.5 Testing the connectivity of resource adapters

You use the connectivity test to verify the correctness of information set up in the resource adapters included in J2EE applications.

You execute the following command to test the connectivity of the resource adapters included in J2EE applications:

Execution format

```
cjtestres [server-name] [-nameserver provider-URL] -name J2EE-application
-name -type rar -resname display-name-of-resource-adapter
```

Example of execution

```
cjtestres -name App1 -type rar -resname DB_Connector_for_Cosminexus_Driver
```

See the following description for the precautions that you must take while testing the connectivity of resource adapters included in J2EE applications:

- Settings for connecting to the database (when the DB Connector is used)
  4.2.4 Testing the connectivity of DB Connector
- Settings for connecting to the other resources (when other resource adapters are used)
  4.3.5 Testing the connectivity of a J2EE resource adapter

For details on the `cjtestres` command, see *cjtestres (execute resource connection test)* in the *uCosminexus Application Server Command Reference Guide*.

5.6 Starting and stopping J2EE applications including resource adapters

When you start a J2EE application, all the resource adapters included in the J2EE application are started. Furthermore, when you stop a J2EE application, all the resource adapters included in the J2EE application are stopped. You cannot control the starting and stopping order of resource adapters.

For details on starting J2EE applications, see *10.2.1 Starting J2EE applications*.

For details on stopping J2EE applications, see *10.2.2 Stopping J2EE applications*.

Note:

When a J2EE application includes resource adapters, and the RAR files are described under the `<module>` tag of `application.xml`, the resource adapters start when the J2EE application starts even if EJB-JAR and WAR files do not reference the resource adapters.

5.7 Referencing the list of resource adapters included in J2EE applications

You can reference the list of resource adapters included in J2EE applications. For resource adapters compliant with Connector 1.5 specifications, you can also reference the information such as the list of connection definition identifiers or list of message listener types.

5.7.1 Referencing the list of resource adapters

You execute the following command to reference the list of resource adapters included in J2EE applications:

Execution format

```
cjlistapp [server-name] [-nameserver provider-URL] -name
J2EE-application-name -type rar [-spec]
```

Example of execution

```
cjlistapp MyServer -name App1 -type rar
```

If you specify the `-spec` option, the version of the connector architecture specification is displayed in the list.

For details on the `cjlistapp` command, see *cjlistapp (list applications)* in the ***uCosminexus Application Server** Command Reference Guide*.

5.7.2 Referencing the list of connection definition identifiers (outbound resource adapter)

For resource adapters compliant with Connector 1.5 specifications, you execute the following command to reference the list of connection definition identifiers of the Outbound resource adapter included in J2EE applications:

Execution format

```
cjlistapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -type rar -resname display-name-of-resource-adapter -outbound
```

Example of execution

```
cjlistapp MyServer -name App1 -type rar -resname account-rar -outbound
```

For details on the `cjlistapp` command, see *cjlistapp (list applications)* in the ***uCosminexus Application Server** Command Reference Guide*.

5.7.3 Referencing the list of message listener types (inbound resource adapter)

For resource adapters compliant with Connector 1.5 specifications, you execute the following command to reference the list of message listener types of the Inbound resource adapter included in J2EE applications:

Execution format

```
cjlistapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -type rar -resname display-name-of-resource-adapter -inbound
```

Example of execution

```
cjlistapp MyServer -name App1 -type rar -resname account-rar -inbound
```

For details on the `cjlistapp` command, see *cjlistapp (list applications)* in the *uCosminexus Application Server Command Reference Guide*.

5.7.4 Referencing the list of property names required for activating the message listener (inbound resource adapter)

For resource adapters compliant with Connector 1.5 specifications, you execute the following command to reference the list of property names required for activating the message listener of the Inbound resource adapter included in J2EE applications:

Execution format

```
cjlistapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -type rar -resname display-name-of-resource-adapter -listenertype me
ssage-listener-type-name
```

Example of execution

```
cjlistapp MyServer -name App1 -type rar -resname account-rar -listenertyp
e javax.jms.MessageListener
```

For details on the `cjlistapp` command, see *cjlistapp (list applications)* in the *uCosminexus Application Server Command Reference Guide*.

5.8 Displaying and deleting the list of connection pools

Display the information and connection state (used or unused) of the connection pools of resource adapters included in J2EE applications. Moreover, delete resource adapter connections as and when required.

5.8.1 Displaying the connection pool state

You execute the following command to display the information and state of connection pools for resource adapters included in J2EE applications:

Execution format

```
cjlistpool [server-name] [-nameserver provider-URL] -name J2EE-application
-name -resname display-name-of-resource-adapter
```

Example of execution

```
cjlistpool MyServer -name App1 -resname Rar1
```

For details on the `cjlistpool` command, see *cjlistpool (list connection pools)* in the *uCosminexus Application Server Command Reference Guide*.

Reference note

For resource adapters compliant with Connector 1.5 specifications, specify *display-name-of-resource-adapter* in the `-resname` option in the following format:

display-name-of-resource-adapter!*connection-definition-identifier*

5.8.2 Deleting connection pools

You execute the following command to delete the connection pools of resource adapters included in J2EE applications as and when required. Note that the command will delete all the unused connection pools.

Execution format

```
cjclearpool [server-name] [-nameserver provider-URL] -type connector -nam
e J2EE-application-name -resname display-name-of-resource-adapter
```

Example of execution

```
cjclearpool MyServer -type connector -name App1 -resname Rar1
```

For details on the `cjclearpool` command, see *cjclearpool (delete connection in connection pool)* in the *uCosminexus Application Server Command Reference Guide*.

Reference note

For resource adapters compliant with Connector 1.5 specifications, specify *display-name-of-resource-adapter* in the `-resname` option in the following format:

display-name-of-resource-adapter!*connection-definition-identifier*

6 Settings for J2EE Resources other than Resource Adapters

This chapter describes how to specify the settings for the J2EE resources other than resource adapters by using the server management commands.

6.1 Overview of settings for J2EE resources other than resource adapters

This section provides an overview of settings for the following J2EE resources:

- Settings for JavaBeans resources
- Settings for mail configuration
- Common settings for J2EE resources

6.1.1 Overview of settings for JavaBeans resources

The following table describes the settings required for using the JavaBeans resources.

Table 6–1: Overview of settings for JavaBeans resources

Settings	Contents	Reference
Importing JavaBeans resources	Import JavaBeans resources and set up the values of the HITACHI JavaBeans Resource properties.	*6.2.1*
Defining the HITACHI JavaBeans Resource Properties	Specify the information required for executing JavaBeans resources.	*6.2.2*
Starting the JavaBeans resources	Start JavaBeans resources.	*6.2.3*
Stopping the JavaBeans resources	Stop JavaBeans resources.	*6.2.4*
Deleting the JavaBeans resources	Delete JavaBeans resources. Execute this setting for switching JavaBeans resources, as and when required.	*6.2.5*
Displaying the state of JavaBeans resources	Display the state of all the imported JavaBeans resources.	*6.2.6*

6.1.2 Overview of mail configuration settings

This setting is required for connecting to the SMTP server by using JavaMail. Set up the information that you want to enter in the mail server and the mail header.

Table 6–2: Overview of mail configuration settings

Settings	Contents	Reference
Creating a new mail configuration	Create a new mail configuration.	*6.3.1*
Defining the mail configuration properties	Set up the following information for the mail configuration: • Host name or IP address of the SMTP mail server • Information to be entered in the mail header	*6.3.2*
Testing the mail configuration for connectivity	Verify that the contents set up in the mail configuration are correct.	*6.3.3*
Deleting the mail configuration	Delete the created mail configuration.	*6.3.4*
Copying the mail configuration	Copy the mail configuration properties.	*6.3.5*

6.1.3 Overview of common settings for J2EE resources

These settings are common to J2EE resources.

Table 6–3: Overview of common settings for J2EE resources

Settings	Contents	Reference
Referencing the list of J2EE resources	Reference the list of following J2EE resources: • J2EE resources included in EJB-JAR files • J2EE resources included in WAR files • Mail configuration	*6.4*
Referencing and changing the J2EE resource names registered in the JNDI name space	Reference the names of J2EE resources registered in the JNDI name space, and add an optional name as and when required.	*6.5*

6.2 Settings for JavaBeans resources

The settings for JavaBeans resources enable the usage of the JavaBeans resources.

To set up JavaBeans resources, you first reference the template files of the HITACHI JavaBeans Resource Property File, and then edit the HITACHI JavaBeans Resource Property File. The *template files of the HITACHI JavaBeans Resource Property File* (`jb_template.xml`) are stored in the following directory:

In Windows:

```
Cosminexus-installation-directory\CC\admin\templates\
```

In UNIX:

```
/opt/Cosminexus/CC/admin/templates/
```

Use the following procedures to set up JavaBeans resources:

To set up a new JavaBeans resource

1. Import a JavaBeans resource.
2. Start the JavaBeans resource.

To change the attributes of the HITACHI JavaBeans Resource Properties

1. Stop the JavaBeans resource.
2. Change the definition of the HITACHI JavaBeans Resource Properties.
3. Start the JavaBeans resource.

To switch the JavaBeans resources

1. Stop the JavaBeans resource.
2. Re-start the J2EE server.
3. Delete the JavaBeans resource.
4. Import the JavaBeans resource.
5. Start the JavaBeans resource.

6.2.1 Importing JavaBeans resources

You execute the following command to import JavaBeans resources:

Execution format

```
cjimportjb [server-name] [-nameserver provider-URL] -f JAR-file-path -c pa
th-of-the-hitachi-JavaBeans-resource-property-file
```

Example of execution

```
cjimportjb MyServer -f Myjavabeans.jar -c Myjavabeansprop.xml
```

If you specify the directory path with the `-d` option of the `cjimportjb` command, you can also import JavaBeans resources as follows:

- Import JavaBeans resources with a directory configuration and without creating an archive.
- Import all the archived JavaBeans resources existing in the specified directory.

The command specification method is as follows:

Execution format

```
cjimportjb [server-name] [-nameserver provider-URL] -d directory-path -c p
ath-of-the-hitachi-JavaBeans-resource-property-file
```

Example of execution

```
cjimportjb MyServer -d MydirectoryPath -c Myjavabeansprop.xml
```

For details on the `cjimportjb` command, see *cjimportjb (import JavaBeans resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

In the `-d` option of the `cjimportjb` command, specify the top level of the directory you want to import. Moreover, all the files existing in the specified directory will be imported, and therefore, make sure that only the required files are included in the specified directory.

6.2.2 Defining the HITACHI JavaBeans Resource Properties

Define the HITACHI JavaBeans Resource Properties.

For an overview of property setting procedures, see *3.5 Property settings using the property file*. This subsection describes the definition of the following HITACHI JavaBeans Resource Properties:

(1) Property file to be edited

HITACHI JavaBeans Resource Property File

(2) Acquiring the property file to be edited and setting up the attributes

- Acquiring the property file
  Execute the following command to acquire a HITACHI JavaBeans Resource Property File:

  Execution format

  ```
  cjgetjbprop [server-name] [-nameserver provider-URL] -resname display-n
  ame-of-JavaBeans-resource -c path-of-the-hitachi-JavaBeans-resource-pro
  perty-file
  ```

  Example of execution

  ```
  cjgetjbprop MyServer -resname MyJavaBeansName -c MyJavaBeansProp.xml
  ```

- Setting up the attributes
  You execute the following command to apply the values of the HITACHI JavaBeans Resource Property File:

Execution format

```
cjsetjbprop [server-name] [-nameserver provider-URL] -resname display-n
ame-of-JavaBeans-resource -c path-of-the-hitachi-JavaBeans-resource-pro
perty-file
```

Example of execution

```
cjsetjbprop MyServer -resname MyJavaBeansName -c MyJavaBeansProp.xml
```

(3) Attribute settings to be edited

The property settings for the JavaBeans resources are as follows:

- Runtime properties
- Optional name information

(a) Runtime properties

The following table describes the settings for the runtime properties (`<property>` tag) of the JavaBeans resources:

Item	Corresponding tags
Property name	<property-name>
Property data type	<property-type>
Property value	<property-value>

Repeat the above settings for all the properties to be defined.

You specify the method names of the `set` method and `get` method of the JavaBeans resources in the settings of the property name (`<property-name>`). For details on the settings of the property name (`<property-name>`), see *4.1.12 Properties that you can specify for the <property> tag* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Optional name information

Specify the optional name (`<optional-name>`) in the optional name information (`<resource-env-external-property>` tag) of JavaBeans resources.

For details on how to use the optional names of JavaBeans resources, see *6.5 Referencing and changing the J2EE resource names registered in the JNDI name space*.

6.2.3 Starting the JavaBeans resources

You execute the following command to start the JavaBeans resources:

Execution format

```
cjstartjb [server-name] [-nameserver provider-URL] -resname display-name-o
f-JavaBeans-resource
```

Example of execution

```
cjstartjb MyServer -resname javabeansname
```

For details on the `cjstartjb` command, see *cjstartjb (start JavaBeans resource)* in the *uCosminexus Application Server Command Reference Guide*.

6.2.4 Stopping the JavaBeans resources

You execute the following command to stop JavaBeans resources:

Execution format

```
cjstopjb [server-name] [-nameserver provider-URL] -resname display-name-of
-JavaBeans-resource
```

Example of execution

```
cjstopjb MyServer -resname javabeansname
```

For details on the `cjstopjb` command, see *cjstopjb (stop JavaBeans resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

Before stopping a JavaBeans resource, you stop all the applications that are using the JavaBeans resource you want to stop.

6.2.5 Deleting the JavaBeans resources

Delete imported JavaBeans resources.

Preparation

Before deleting a JavaBeans resource, you stop the JavaBeans resource, and then restart the J2EE server.

You execute the following command to delete JavaBeans resources:

Execution format

```
cjdeletejb [server-name] [-nameserver provider-URL] -resname display-name
-of-JavaBeans-resource
```

Example of execution

```
cjdeletejb MyServer -resname MyJavaBeans
```

For details on the `cjdeletejb` command, see *cjdeletejb (delete JavaBeans resource)* in the *uCosminexus Application Server Command Reference Guide*.

6.2.6 Displaying the state of JavaBeans resources

Display the names and states (running or terminated) of all the JavaBeans resources.

You execute the following command to display the state of JavaBeans resources:

Execution format

```
cjlistjb [server-name] [-nameserver provider-URL]
```

Example of execution

```
cjlistjb MyServer
```

For details on the `cjlistjb` command, see *cjlistjb (list JavaBeans resources)* in the *uCosminexus Application Server Command Reference Guide*.

6.3 Settings for mail configuration

The mail configuration settings are required for connecting to the SMTP server by using JavaMail.

6.3.1 Creating a new mail configuration

Create a new mail configuration.

You execute the following command to create a new mail configuration based on the mail property file:

Execution format

```
cjsetresprop [server-name] [-nameserver provider-URL] -type mail -resname
display-name-of-mail -c path-of-mail-property-file
```

Example of execution

```
cjsetresprop MyServer -type mail -resname Mail -c MailProp.xml
```

For details on the `cjsetresprop` command, see *cjsetresprop (set Property of resource)* in the *uCosminexus Application Server Command Reference Guide*. For property files, see *4.3 HITACHI Mail Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

6.3.2 Defining the mail configuration properties

Define the mail configuration properties. You can change the contents that are defined when a new mail configuration is created.

For an overview of property setting procedures, see *3.5 Property settings using the property file*. The definition of the mail configuration properties is as follows:

(1) Property file to be edited

Mail property file

(2) Acquiring the property file to be edited and setting the attributes

- Acquiring the property file
  Execute the following command to acquire a mail property file:

  Execution format

  ```
  cjgetresprop [server-name] [-nameserver provider-URL] -type mail -resna
  me display-name-of-mail -c path-of-mail-property-file
  ```

  Example of execution

  ```
  cjgetresprop MyServer -type mail -resname Mail -c MailProp.xml
  ```

- Setting the attributes
  Execute the following command to apply a mail property file:

Execution format

```
cjsetresprop [server-name] [-nameserver provider-URL] -type mail -resna
me display-name-of-mail -c path-of-mail-property-file
```

Example of execution

```
cjsetresprop MyServer -type mail -resname Mail -c MailProp.xml
```

(3) Attribute settings to be edited

The following table describes the settings for the mail configuration properties:

Items	Corresponding tags
E-mail address of the E-mail sender	<from>
Host name or IP address of the SMTP mail server	<server>
Optional name information#	<resource-external-property>

#

Specify the following items in the optional name information (`<resource-external-property>`):

Items in optional name information	Mandatory	Corresponding tags
Optional name of resources	Y	<optional-name>
Resource authentication method	O	<res-auth>
Scope of resource sharing	O	<res-sharing-scope>

Legend:

Y: Mandatory

O: Optional

For details on the property settings, see *4.3 HITACHI Mail Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

6.3.3 Testing the mail configuration for connectivity

Check the mail server connection to verify whether the contents set up in the mail configuration are correct.

You execute the following command to test the connectivity of the mail configuration:

Execution format

```
cjtestres [server-name] [-nameserver provider-URL] -type mail -resname dis
play-name-of-mail-configuration
```

Example of execution

```
cjtestres MyServer -type mail -resname Mymail1
```

For details on the `cjtestres` command, see *cjtestres (execute resource connection test)* in the *uCosminexus Application Server Command Reference Guide*.

6.3.4 Deleting the mail configuration

You execute the following command to delete the mail configuration:

Execution format

```
cjdeleteres [server-name] [-nameserver provider-URL] -type mail -resname d
isplay-name-of-mail-configuration
```

Example of execution

```
cjdeleteres MyServer -type mail -resname Mail
```

For details on the `cjdeleteres` command, see *cjdeleteres (delete resource)* in the *uCosminexus Application Server Command Reference Guide*.

6.3.5 Copying the mail configuration

You execute the following command to copy the mail configuration properties:

Execution format

```
cjcopyres [server-name] [-nameserver provider-URL] -type mail -src display
-name-of-mail-configuration-at-copy-source -dst display-name-of-mail-confi
guration-at-copy-destination
```

Example of execution

```
cjcopyres MyServer -type mail -src Mail -dst Mail2
```

For details on the `cjcopyres` command, see *cjcopyres (copy resource)* in the *uCosminexus Application Server Command Reference Guide*.

6.4 Referencing the list of J2EE resources

Reference the following list of J2EE resources:

- Imported EJB-JAR file and J2EE resources included in EJB-JAR file
- Imported WAR file and J2EE resources included in WAR file
- Imported mail configuration

Referencing the list of J2EE resources included in the EJB-JAR file

You execute the following command to reference the list of J2EE resources included in EJB-JAR files:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type ejb [-resname
display-name-of-EJB-JAR-file]
```

- If you specify the `-resname` option, the list of resources (SessionBeans, EntityBeans, and MessageDrivenBeans) included in a specific EJB-JAR file will be displayed.
- If you do not specify the `-resname` option, the list of imported EJB-JAR files will be displayed.

Example of execution

```
cjlistres MyServer -type ejb
```

Referencing the list of J2EE resources included in WAR files

You execute the following command to reference the list of J2EE resources included in a WAR file:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type war [-resname
display-name-WAR-file]
```

- If you specify the `-resname` option, the list of resources (servlets, JSPs, and filters) included in a specific WAR file will be displayed.
- If you do not specify the `-resname` option, the list of imported WAR files will be displayed.

Example of execution

```
cjlistres MyServer -type war
```

Referencing the list of mail configurations

You execute the following command to reference the list of imported mail configurations:

Execution format

```
cjlistres [server-name] [-nameserver provider-URL] -type mail
```

Example of execution

```
cjlistres MyServer -type mail
```

For details on the `cjlistres` command, see *cjlistres (list resources)* in the *uCosminexus Application Server Command Reference Guide*.

6.5 Referencing and changing the J2EE resource names registered in the JNDI name space

Change the J2EE resource names registered in the JNDI name space.

The optional names are also added with the J2EE resource names. By adding the optional names, you can reference J2EE resources from the JNDI name space using the optional names. This functionality is called the *user-specified namespace functionality*.

For the JNDI name space of resource adapters, see *2.3 Binding and lookup of objects to the JNDI name space* in the *uCosminexus Application Server Common Container Functionality Guide*. For the user-specified name space functionality, see *2.6 Adding optional names to Enterprise Beans or J2EE resources (user-specified name space functionality)* in the *uCosminexus Application Server Common Container Functionality Guide*.

You can add optional names in the following property files:

Property file	Corresponding tag names
Mail property file	<resource-external-property>
HITACHI JavaBeans Resource Property File	<resource-env-external-property>

For details on editing the `<resource-external-property>` tag of the mail property file, see the settings of the optional name information in *6.3.2 Defining the mail configuration properties* and for details on editing the `<resource-env-external-property>` tag of the HITACHI JavaBeans Resource Property File, see the settings of the optional name information in *6.2.2 Defining the HITACHI JavaBeans Resource Properties*.

Note:

If a J2EE application is running, you cannot stop and delete a J2EE resource with the specified optional name. Stop all the J2EE applications running on the J2EE server.

7 Creating J2EE Applications

This chapter describes how to create J2EE applications with server management commands.

7.1 Overview of creating J2EE applications

Creating a J2EE application refers to the process of using an Enterprise Bean (EJB-JAR) and servlet created in an application development environment and a JSP (WAR) to create a J2EE application (EAR). In a J2EE application, you can also include resource adapters that you want to use in that J2EE application.

The following table describes the operations required for creating a J2EE application.

Table 7–1: Operations to be executed for creating a J2EE application

Items to be executed	Contents	Reference
Importing the Enterprise Beans (EJB-JAR)	Operation required when the Enterprise Beans (EJB-JAR) are included in the configuration of the J2EE application. Import the Enterprise Beans into the J2EE server.	*7.2.1*
Importing the servlet and JSP (WAR)	Operations required when the servlets and JSPs (WAR) are included in the configuration of the J2EE application. Import the servlets and JSPs into the J2EE server.	*7.2.2*
Importing resource adapters (RAR files)	When resource adapters (RAR) are included in J2EE applications, import the included resource adapters into the J2EE server.	*7.2.3*
Creating a new J2EE application	Create an EAR file on the basis of the imported EJB-JAR and WAR files. Include RAR in J2EE applications as and when required.	*7.2.4*
Adding library JAR files to the J2EE application	Add the library JAR files (file extension is the lower case `.jar`) to the J2EE application.	*7.2.5*
Referencing the list of library JAR files	Reference the list of library JAR files included in the J2EE application.	*7.2.6*
Deleting the library JAR files from the J2EE application	Delete the library JAR files included in the J2EE application.	*7.2.7*
Setting a reference library in the J2EE application	Set the reference library in the J2EE application.	*7.2.8*
Property settings of J2EE applications	Set the properties for the created J2EE application.	*Chapter 9*
Executing J2EE applications	Execute the J2EE application for which setup is complete. Terminate the applications based on the operations.	*Chapter 10*

To create J2EE applications with `cosminexus.xml`, you include `cosminexus.xml`, created in advance, along with EJB-JAR and WAR files in J2EE applications, and then use.

The library JAR and reference library are *common libraries* and you can reference these libraries from each module of the J2EE application.

The library JAR and reference library have the following features:

- *Library JAR*
    - The J2EE application includes a JAR (file extension is the lower case `.jar`).
    - When used in multiple J2EE applications, a library JAR needs to be added in each J2EE application.
    - The library JAR cannot include a class file. To use a class file, the class file needs to be compiled in a JAR file format.
- *Reference library*
    - The reference library specifies the absolute path of the library file to be referenced.

- When used in multiple J2EE applications, you can use the same reference library by simply specifying the same reference.
- The reference library can handle JAR files as well as class files.

When including the following classes in a common library that can be referenced from each module in a J2EE application, you must use a library JAR to include the classes: classes coded in an `ejb-jar.xml` or `web.xml` file, methods (arguments, return values, and exceptions), classes# that are loaded for reading annotation information, and classes required for the reference solution of the included classes. When these classes are used as a reference library, `java.lang.NoClassDefFoundError` or `java.lang.ClassNotFoundException` occurs when importing applications, and an attempt to acquire the annotation information fails (`KDJE42380-W` is output) or an attempt to import fails.

#

For details on the classes loaded for reading the annotation information, see *14.3 Classes to be loaded and the class path required for loading* in the *uCosminexus Application Server Common Container Functionality Guide*.

7.2 Details of creating J2EE applications

7.2.1 Importing the Enterprise Beans (EJB-JAR)

Before executing the J2EE application by using the EJB components, import the Enterprise Beans (EJB-JAR). You can import EJB-JARs with DDs conforming to EJB 1.1, 2.0, 2.1, or 3.0 specifications, or EJB-JARs with annotations.

Note that various EJB sample programs are stored in the following directories:

In Windows

Cosminexus-installation-directory`\CC\examples\`

In UNIX

`/opt/Cosminexus/CC/examples/`

For an EJB-JAR with a DD, confirm that the DD conforms to the DTD and then import the EJB-JAR. If an EJB-JAR with a DD not conforming to the DTD specifications is imported, a message will be output. For details on the output messages, see the manual *uCosminexus Application Server Messages*.

Execute the following command to import EJB-JAR:

Execution format

```
cjimportres [server-name] [-nameserver provider-URL] -type ejb -f path-of
-EJB-JAR-file
```

Example of execution

```
cjimportres MyServer -type ejb -f account.jar
```

For details on the `cjimportres` command, see *cjimportres (import resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- If you import an EJB-JAR with a version 1.1 DD, the DD version in the J2EE server will change to version 2.0.
- You cannot import an EJB-JAR with a DD not conforming to EJB 1.1, 2.0, 2.1, or 3.0 specifications.
- Do not import EJB-JAR with the name `hitachi-runtime.jar`.
- The JAR file name specified when importing is used as the directory name of the working directory. Also, in the JAR file, if there is a class that implements java.rmi.Remote, (for example, the home interface), then based on the definition in the `usrconf.properties` file, a stub is generated in the working directory when deploying or starting the application. In the case of default settings, a stub will be created for the remote and home interfaces coded in `ejb-jar.xml`. When `app` is specified in the ejbserver.deploy.stub.generation.scope key of the `usrconf.properties` file, the stubs will be generated for all the interfaces that implement the remote interface included in the JAR file.

  Considering the above points, specify a JAR file name and the package name or the class name of the class that implements the java.rmi.Remote, so that the path length of the working directory does not reach the upper limit specified for the platform.

  For details on estimating the path length of the working directory, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*.

7.2.2 Importing the servlet and JSP (WAR)

Import the servlet and JSP (WAR) before using Web components in the J2EE server mode to execute a J2EE application.

Note that you can import a WAR with a DD conforming to Servlet 2.2, 2.3, 2.4, or 2.5 specifications, or WAR with annotations. Confirm that the DD conforms to the DTD specifications and then import the WAR. If you import a WAR with a DD not conforming to the DTD specifications, a message will be output. For details on the output messages, see the manual *uCosminexus Application Server Messages*.

Execute the following command to import a WAR file:

Execution format

```
cjimportres [server-name] [-nameserver provider-URL] -type war -f path-of
-WAR-file
```

Example of execution

```
cjimportres MyServer -type war -f account.war
```

For details on the `cjimportres` command, see *cjimportres (import resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- If you import a WAR with a version 2.2 DD, the DD version in the J2EE server will change to version 2.3.
- You cannot import a WAR with a DD not conforming to Servlet 2.2, 2.3, 2.4, or 2.5 specifications.
- The WAR file name specified when importing is used as the directory name of the working directory. When deploying or starting the application, the files in the WAR file are deployed in the working directory. Also, in the WAR file, if there is a class that implements java.rmi.Remote (for example, the home interface), then based on the definition in the `usrconf.properties` file, a stub is generated in the working directory when deploying or starting the application. When `app` is specified in the ejbserver.deploy.stub.generation.scope key of the `usrconf.properties` file, the stubs will be generated for all the interfaces that implement the remote interface included in the Jar file.

  Considering the above points, specify a WAR file name and the package name or the class name of the WAR file and the package name and the class name of the class that implements java.rmi.Remote, so that the path length of the working directory does not reach the upper limit specified for the platform.

  For details on estimating the path length of the working directory, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*.
- Do not import a WAR with the name `hitachi-runtime.jar`.

7.2.3 Importing resource adapters (RAR files)

Import resource adapters (RAR) to include in J2EE applications. You can include resource adapters in a J2EE application to use resource adapters from the Enterprise Beans (EJB-JAR), servlets, and JSPs (WAR) of the same J2EE application.

For details on the resource adapters which you can import, see *5.1.1 Available resource adapters*.

You execute the following command to import RAR files:

Execution format

```
cjimportres [server-name] [-nameserver provider-URL] -type rar -f path-of
-RAR-file
```

Example of execution

```
cjimportres MyServer -type rar -f account.rar
```

For details on the `cjimportres` command, see *cjimportres (import resource)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

You confirm that RAR files have a DD compliant with Connector version 1.0 or later specifications, and then start importing the RAR files. You cannot import RAR with a DD that is not compliant with J2EE Connector version 1.0 or later specifications.

7.2.4 Creating a new J2EE application

Create a J2EE application on the basis of the imported EJB-JAR and WAR files. Moreover, add RAR files that you want to use with EJB-JAR and WAR files.

(1) Adding the EJB-JAR files

Execute the following command to add an EJB-JAR file. If the J2EE application does not exist, a new J2EE application will be created.

Execution format

```
cjaddapp [server-name] [-nameserver provider-URL] -type ejb -name J2EE-app
lication-name -resname EJB-JAR-display-name
```

Example of execution

```
cjaddapp MyServer -type ejb -name adder -resname adder
```

For details on the `cjaddapp` command, see *cjaddapp (add resource)* in the *uCosminexus Application Server Command Reference Guide*.

(2) Adding the WAR files

Execute the following command to add a WAR file. If the J2EE application does not exist, a new J2EE application will be created.

Execution format

```
cjaddapp [server-name] [-nameserver provider-URL] -type war -name J2EE-app
lication-name -resname WAR-display-name
```

Example of execution

```
cjaddapp MyServer -type war -name adder -resname adder_war
```

For details on the `cjaddapp` command, see *cjaddapp (add resource)* in the *uCosminexus Application Server Command Reference Guide*.

(3) Adding RAR files

You execute the following command to add RAR files. If a J2EE application does not exist, a new J2EE application will be created:

Execution format

```
cjaddapp [server-name] [-nameserver provider-URL] -type rar -name J2EE-app
lication-name -resname RAR-display-name
```

Example of execution

```
cjaddapp MyServer -type rar -name adder -resname account-rar
```

For details on the `cjaddapp` command, see *cjaddapp (add resource)* in the *uCosminexus Application Server Command Reference Guide*.

(4) Notes

- When creating the J2EE application, the name corresponding to the *J2EE-APP-name* of JNDI name space (HITACHI_EJB/SERVERS/*server-name*/EJB/*J2EE-APP-name*/*Enterprise-Bean-name*) of the EJB home object is automatically allocated. For details on the JNDI name space, see *2.3 Binding and lookup of objects to the JNDI name space* in the *uCosminexus Application Server Common Container Functionality Guide*.
- For the applications in the exploded archive format, you cannot add EJB-JAR files, WAR files, and RAR files.
- You cannot add EJB-JAR files without the extension `.jar`, WAR files without the extension `.war`, and RAR files without the extension `.rar` to applications where `application.xml` is omitted.
- You cannot add a filter to WAR files where `web.xml` is omitted.
- In the J2EE application name, specify only single-byte alphanumeric characters (0 to 9, A to Z, a to z) or the following special characters:
  Plus sign (+), hyphen (-), period (`.`), caret symbol (`^`), and underscore (_)
- Once you have created the J2EE application, you cannot change the name of the J2EE application.

7.2.5 Adding library JAR files to the J2EE application

To add the library JAR files to the J2EE application, execute the following command:

Execution format

```
cjimportlibjar [server-name] [-nameserver provider-URL] -name J2EE-applica
tion-name -f library-JAR-name
```

Example of execution

```
cjimportlibjar MyServer -name App1 -f applib.jar
```

For details on the `cjimportlibjar` command, see *cjimportlibjar (import library JAR)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- The JAR file name specified when importing is used as the directory name in the working directory. Specify the JAR file name so that the path length of the working directory does not reach the upper limit specified for the platform. For details on estimating the path length of the working directory, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*.
- Specify the lower case `.jar` as the extension of the library JAR file.
- You cannot add or delete library JARs in the J2EE applications that are in the exploded archive format.
- Do not import a library JAR with the name `hitachi-runtime.jar`.

7.2.6 Referencing the list of library JAR files

To reference the list of library JAR files imported into a J2EE application, execute the following command:

Execution format

```
cjlistlibjar [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name
```

Example of execution

```
cjlistlibjar MyServer -name App1
```

For details on the `cjlistlibjar` command, see *cjlistlibjar (list library JARs)* in the *uCosminexus Application Server Command Reference Guide*.

7.2.7 Deleting the library JAR files from the J2EE application

To delete the library JAR files imported into a J2EE application, execute the following command:

Execution format

```
cjdeletelibjar [server-name] [-nameserver provider-URL] -name J2EE-applica
tion-name -f library-JAR-file-name
```

Example of execution

```
cjdeletelibjar MyServer -name App1 -f applib.jar
```

For details on the `cjdeletelibjar` command, see *cjdeletelibjar (delete library JAR)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

You cannot add or delete a library JAR in J2EE applications that are in the exploded archive format.

7.2.8 Setting a reference library in the J2EE application

You set a reference library in a J2EE application.

Set the reference library of a J2EE application in the J2EE application attribute file. For details on how to set the properties, see *3.5 Property settings using the property file*.

(1) Attribute file to be edited

Application attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire the application attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
ation-name -c path-of-application-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name App1 -c C:\home\app1.xml
```

- Setting the attributes
  Execute the following command to apply the values of the application attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
ation-name -c path-of-application-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name App1 -c C:\home\app1.xml
```

(3) Attribute settings to be edited

Specify the class path (`<classpath>`) of the specification (`<ref-libraries>`) of the library JAR to be referenced.

To specify the reference library class that is set up here in the property file, you must copy that reference library file onto the machine that executes server management commands, and then specify the path of the copied reference library file for the `USRCONF_JVM_CLASSPATH` in the option definition file used for server management commands.

8 Importing and Exporting J2EE Applications

This chapter describes how to import and export J2EE applications with the server management commands.

8.1 Importing the J2EE applications

You can import J2EE applications containing DDs or J2EE applications with annotations. Import J2EE applications containing DDs after ensuring that the DDs of EAR, EJB-JAR, WAR, and RAR comply with the DTD.

J2EE applications to be imported are in the following formats:

- J2EE applications that are in archive format
  The J2EE applications in the archive format are the J2EE applications containing the components of EJB-JAR, WAR, and RAR files in the working directory of the J2EE server.
- J2EE applications that are in exploded archive format
  The J2EE applications in the exploded archive format are the J2EE applications that use the configuration files external to the J2EE server as logical J2EE applications. When switching the J2EE applications by reloading, you need to import the J2EE applications in this format. For switching the J2EE applications, see *10.5 Switching between J2EE applications*.

8.1.1 Importing the J2EE applications that are in archive format

Execute the following command to import J2EE applications that are in archive format:

Execution format

```
cjimportapp [server-name] [-nameserver provider-URL] -f EAR-file-path
```

Execution example

```
cjimportapp MyServer -f C:\home\bank.ear
```

For details on the `cjimportapp` command, see *cjimportapp (import J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

(1) Precautions related to runtime information

The term *runtime information* references to the Cosminexus J2EE server-specific information. The runtime information includes the following information.

Table 8–1: List of runtime information

Items	Information
Status of J2EE application	Information about start or stop status
Various settings of J2EE applications	• Runtime properties of Enterprise Beans • Mapping of each reference and J2EE resource • Mapping of CMP field database
Auto-generated implementation class	• EJB home object • EJB object
stub class, tie class of the remote interface	• EJB home object • EJB object

The following are the precautions to be taken when importing the EAR files that include the runtime information:

- The status of a J2EE application (when the J2EE application was exported) is restored. Consequently, if a running J2EE application is imported, the execution of the J2EE application starts immediately.
- The various settings of the J2EE applications are restored. If you create a J2EE resource adapter, JDBC data source, or JavaMail configuration with the same name as that of the exported J2EE server on the J2EE server to be imported, then the reference is resolved when importing the J2EE application. Therefore, when importing a running J2EE application, create a J2EE resource adapter, JDBC data source, or JavaMail configuration beforehand by using the same name as that of the J2EE server that was exported. Furthermore, import the J2EE application when the J2EE resource adapter is running.
- The auto-generated implementation class, remote interface stub class, and remote interface tie class included in an EAR file with runtime information are re-used; however, these classes are re-created when you import an EAR file that was exported with an earlier version.
- Ensure that the execution environment information file is not changed.

(2) Precautions related to DDs of EARs

You cannot import an EAR containing DDs not conforming to Java EE specifications.

If the DD version is old, the version is changed to as follows.

Table 8–2: Changing DD version

DD	Version before importing	Version after importing
`application.xml`	1.2	1.4
	1.3	
`ejb-jar.xml`	1.1	2.0
`web.xml`	2.2	2.3

(3) Other precautions

- Do not include files with the following names when the files to be imported are not the J2EE applications that are exported from aJ2EE server:
    - `hitachi-runtime.jar`
    - `rar.properties`
    - `fileinfo.properties`
    - `META-INF/hitachi-ra.xml`
- When a J2EE application is started, a name that corresponds to *J2EE-APP-name* in the JNDI name space of the EJB home object (HITACHI_EJB/SERVERS/*server-name*/EJB/*J2EE-APP-name*/*Enterprise-Bean-name*) is automatically allocated. For details on the JNDI name space, see *2.3 Binding and lookup of objects to the JNDI name space* in the *uCosminexus Application Server Common Container Functionality Guide*.
- The EJB-JAR file name in an EAR file is used as a directory name in the working directory. If there is a class that implements java.rmi.Remote (for example, the home interface) in the EJB-JAR file, then based on the definition in the `usrconf.properties` file, a stub is generated in the working directory when deploying or starting the application. In the case of default settings, a stub is generated for the <remote> and <home> interfaces coded in ejb-jar.xml. If you specify `app` in the ejbserver.deploy.stub.generation.scope key of the `usrconf.properties` file, the stubs will be generated for all the interfaces that implement the remote interface included in the EJB-JAR file.

Considering the above points, specify an EJB-JAR file name and the package name and the class name of the class that implements java.rmi.Remote so that the path length of the working directory is within the limit specified for the platform.

For details on estimating the path length of the working directory, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*.

- The WAR file name in an EAR file is used as a directory name in the working directory. The files in a WAR file are deployed in the working directory. If a WAR file contains a class that implements java.rmi.Remote (for example, the home interface), then based on the definition in the `usrconf.properties` file, a stub is generated in the working directory when deploying or starting an application.

  If you specify `app` in an ejbserver.deploy.stub.generation.scope key of the `usrconf.properties` file, the stubs will be generated for all the interfaces that implement the remote interface of the WAR file.

  Considering the above points, specify the WAR file name, the package name and class name of the WAR file, and the package name and class name of the class that implements java.rmi.Remote, so that the path length of the working directory does not reach the limit specified for the platform.
- The library JAR file name in an EAR file is used as a directory name in the working directory. Specify the JAR file name so that the path length of the working directory does not reach the limit specified for the platform.
- You cannot import the files when the display names of EJB-JAR files and WAR files in EAR files are the same. When a display name is a blank character or is not specified, a character string where the file name is changed is specified. You cannot import the file even when this value is duplicated.
- If an error occurs in the runtime information of a J2EE application that contains imported runtime information, importing of the 2EE application fails. At this time, the J2EE application is deleted without being started. To avoid the deletion of the J2EE application, specify the `-nodelete` option of the `cjimportapp` command to import the J2EE application. In such a case, the J2EE application can be imported without being deleted but since the application does not start automatically, eliminate the errors as and when required and then start the J2EE application.
- The rules to convert the display name and the directory name when the applications to be imported include `application.xml` are as follows:
  - The value of the <`display-name`> tag in `application.xml` becomes the display name.
  - If the <`display-name`> tag does not have a value, a string converting the EAR file name specified in the `-f` option is considered as the display name.

    Note that if the file name contains characters other than general-purpose characters and numbers, the characters are substituted by underscores (_).

    If the characters to be substituted are continuous, the characters are combined into one underscore (_).
  - The name excluding the extension from the path name described in the <`module`> tag in `application.xml` becomes the same name as the EJB-JAR directory name or WAR directory name.
- The rules to convert the display name and the directory name when the applications to be imported exclude `application.xml` are as follows:
  - The converted string excluding the extension from the file name specified in the `-f` option becomes the display name.

    Note that the characters other than one-byte alphanumeric characters (0 to 9, A to Z, and a to z) and underscores (_) are substituted by underscores (_).
  - For file names that begin with a period (.), the period (.) is not removed and the converted character string becomes the display name.
  - The EJB-JAR directory name will end with `_jar` and the WAR directory name will end with `_war`.

8.1.2 Importing the J2EE applications that are in exploded archive format

J2EE applications that are in exploded archive format do not contain a class file and JAR file of each component in the working directory of J2EE server. These applications contain only the DDs and the path information of the exploded archive and are the J2EE applications with EAR files or ZIP files deployed in the exploded archive path. However, the RAR files included in J2EE applications are stored in the archive format.

You can import the J2EE applications that are in exploded archive format by the following two methods:

- Import an application directory as an application that is in exploded archive format.
- Import a J2EE application as an application that is in exploded archive format.

Important note

When importing an application that is in the exploded archive format, do not add, delete, or overwrite the directories or files under the application directory while a J2EE server is running or a server management command is running.

The following are the procedures to import the applications that are in exploded archive format:

(1) Importing an application directory as an application in exploded archive format

Use the following procedures to register an application directory in the J2EE server:

1. Create an application directory.
   Create an application directory to create a new J2EE application in exploded archive format. For details on the configuration of an application directory, see *15.4.2 Configuring an application directory* in the *uCosminexus Application Server Common Container Functionality Guide*.
2. Import the application directory.
   Register the application directory created in step 1 in the J2EE server.
3. Execute the following command to import an application directory:

(a) Execution format

```
cjimportapp [server-name] -a  application-directory-path
```

(b) Execution example

```
cjimportapp MyServer -a AppDirPath
```

For details on the `cjimportapp` command, see *cjimportapp (import J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

(c) Note

- You cannot execute this command in a remote environment.
- You cannot use the application directory that specifies the `<alt-dd>` tag in `application.xml`, in exploded archive format.

- The rules to convert the display name and the directory name when the applications to be imported include `application.xml` are as follows:
  - A value of the `<display-name>` tag of `application.xml` is the display name.
  - If the `<display-name>` tag does not have a value, a string converting the EAR file name specified in the `-f` option is considered as the display name.

    Note that if the file name contains characters other than general-purpose characters and numbers, the characters are substituted by underscores (_).

    If the characters to be substituted are continuous, the characters are combined into one underscore (_).
  - The name excluding the extension from the path name described in the `<module>` tag of `application.xml` will be the same as the EJB-JAR directory name or WAR directory name.
- The rules to convert the display name and the directory name when the applications to be imported exclude `application.xml` are as follows:
  - Remove the extension from the file name specified in the `-f` option, and thereby the changed character string becomes the display name.

    Note that the characters other than one-byte alphanumeric characters (0 to 9, A to Z, and a to z) and underscores (_) are substituted by underscores (_).
  - For a file name that begins with a period (.), the changed character string without the period (.) removed becomes the display name.
  - The EJB-JAR directory name will end with `_jar` and the WAR directory name will end with `_war`.
- You cannot import an application if a J2EE application that uses the same directory as an application directory exists in the J2EE server.
- When an application directory created by using the `-d` option and deploying an EAR file or ZIP file matches either of the following conditions, you cannot import the application directory with the `-a` option and use it as an exploded-archive application directory.
  - EJB-JAR module name does not end with '.jar'.
  - WAR module name does not end with '.war'.
  - The module name that does not contain an extension is the same as any other module name that does not contain an extension.
  - The module name that does not contain an extension is the same as a directory in an EAR file.
- The DD for various components under the application directory has UTF-8 encoding.
- If the DD version is old, the version is changed to as follows:

Table 8–3: Changing the DD version

DD	Version before importing	Version after importing
`application.xml`	1.2	1.4
	1.3	
`ejb-jar.xml`	1.1	2.0
`web.xml`	2.2	2.3

- When importing an exploded-archive J2EE application, the process searches for the library JAR below the application directories excluding the EJB-JAR and WAR directories. Therefore, when several files exist below the application directories excluding the EJB-JAR and WAR directories, import may take time.

(2) Importing a J2EE application as an application in exploded archive format

Import the EAR file or the EAR file containing the exported runtime information in exploded archive format.

Execute the following command to import an application in exploded archive format:

(a) Execution format

```
cjimportapp [server-name]-f  file-path -d exploded-archive-path
```

The contents of an EAR file or ZIP file are deployed in the exploded archive path.

(b) Execution example

```
cjimportapp MyServer -f App1.zip -d ApplicationDir
```

For details on the `cjimportapp` command, see *cjimportapp (import J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

(c) Note

- The user who starts the J2EE server requires write-permission for the directory specified in the exploded archive path.
- You cannot use a J2EE application with an <alt-dd> tag specified in `application.xml`, in exploded archive format.
- You cannot import an application if a J2EE application that uses the same directory as the exploded archive exists in the J2EE server.
- The other precautions are the same as when importing J2EE applications that are in the archive format. To import J2EE applications that are in the archive format, see the precautions described in *8.1.1 Importing the J2EE applications that are in archive format*.
- The DD for various components under the application directory has UTF-8 encoding.

8.2 Exporting J2EE applications

Execute the following command to export the J2EE applications:

Execution format

```
cjexportapp [server-name] [-nameserver provider-URL] -name J2EE-applicatio
n-name -f EAR-file-path [-raw|-normal]
```

When runtime information is included

Specify -normal (default value).

When runtime information is not included

Specify -raw.

Execution example

```
cjexportapp MyServer -name account -f C:\home\account.zip
```

Runtime information of an EAR file is included in this example.

For details on the `cjexportapp` command, see *cjexportapp (export J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

The following are the precautions to be taken when exporting a J2EE application:

- Precautions related to runtime information
- Precautions related to the DD of EAR files

Precautions related to runtime information

- When exporting a J2EE application, you can select either of the files mentioned below. The default file format contains the runtime information (ZIP format).

File containing runtime information (ZIP format)

These are the ZIP format files containing runtime information. Note that the file containing the runtime information is unique to Cosminexus.

EAR

An EAR file is provided in Java 2 Platform, Enterprise Edition. An EAR does not contain runtime information.

The J2EE applications that are exported in a format containing runtime information are not downward compatible. Do not change the contents of J2EE applications.

In a system that consists of multiple J2EE servers, when JDBC resource settings, database used, and runtime properties of the J2EE applications are to be the same among the J2EE servers, use EAR files that include runtime information. At this time, Hitachi recommends exporting the J2EE applications created in one J2EE server and importing it on the J2EE servers.

When importing a J2EE application created on a J2EE server to a J2EE server on another system or to a J2EE server that has a different execution environment on the same system, or when using the application with other J2EE server products, export the J2EE application by using the EAR file format.

- When exporting an exploded archive format application that is registered in a J2EE server, generate an EAR file or a ZIP file that includes runtime information.
- When exporting an application that is JSP pre-compiled at application start time, the JSP compilation result is included in the exported EAR file. When the EAR file that includes the JSP compilation result is imported to another J2EE server, you can use the compilation result of the JSP file included in the EAR file.

- When an imported J2EE application contains `cosminexus.xml`, the application is exported with `cosminexus.xml` as it is. For details on exporting applications that include `cosminexus.xml`, see *13.3.6 Running an application that includes cosminexus.xml* in the *uCosminexus Application Server Common Container Functionality Guide*.

Precautions related to the DD of EAR Files

- Depending on the J2EE application version, the DD version of EAR is 1.4 or later.
- Depending on the EJB-JAR version, the DD version of EJB-JAR included in EAR is 2.0 or later.
- Depending on the WAR version, the DD version of WAR included in EAR is 2.3 or later.
- The DD for the various components included in an EAR has UTF-8 encoding.

9 Property Settings of J2EE applications

This chapter describes how to set the properties of J2EE applications with the server management commands.

9.1 Overview of the property settings of J2EE applications

The J2EE application property settings imply defining the attributes and operations of the imported J2EE applications. This section explains the following:

- Property settings and attribute files for J2EE applications
- Property settings for Enterprise Beans
- Property settings for servlets and JSPs
- Property settings for J2EE applications (common)

> **Important note**
>
> You cannot change the items specified in annotations. Furthermore, to change application properties that contain `cosminexus.xml`, there are setup methods that use the server management commands and MyEclipce. If you use the redeploy functionality after changing the definition information within `cosminexus.xml` by using server management commands, the information changed by the server management commands is lost.
>
> For details on changing the application properties that contain `cosminexus.xml` by using MyEclipce, see *5.3.2 How to operate cosminexus.xml editor* in the *uCosminexus Application Server Application Development Guide*. Also, for details on the operations of applications that contain `cosminexus.xml`, see *13.3.6 Running an application that includes cosminexus.xml* in the *uCosminexus Application Server Common Container Functionality Guide*.

For details on the property settings and property files for resource adapters included in J2EE applications, see *5.4 Defining resource adapter properties*.

9.1.1 Property settings and attribute files for J2EE applications

For an overview about the procedures for setting the properties, see *3.5 Property settings using the property file*.

You can use the following attribute files to set the J2EE application properties:

- Specific attribute files
  Set the properties for each specific attribute file of the components that form the J2EE applications.
- HITACHI Application Integrated Property File
  Compile the contents of specific attribute files and set the properties.

In the J2EE application property settings, you can use specific attribute files and HITACHI Application Integrated Property Files individually as well as collectively.

This chapter mainly explains how to set the properties by using the specific attribute files. For details on setting the properties by using the HITACHI Application Integrated Property File, see *9.2 Property settings using the HITACHI Application Integrated Property File*.

9.1.2 Property settings of an Enterprise Bean

When you create and customize J2EE applications containing Enterprise Beans, specify the settings below when required.

The following table describes the property settings for the Enterprise Beans:

Table 9–1: Items to be edited in the Enterprise Bean properties

Items to be edited		Contents	Mandatory	Reference
Defining Enterprise Bean references		Define the references to the following Enterprise Beans forming a J2EE application: • Session Bean • Entity Bean • Message-driven Bean	O	*9.3*
	Defining other Enterprise Bean references	Define the references to the other Enterprise Beans.	O	*9.3.1*
	Defining mail configuration references	Define the references to the mail servers.	O	*9.3.2*
	Defining resource adapter references	Define the references to the resource adapters.	O	*9.3.3*
	Defining resource environment references	Define the references to the resource environment.	O	*9.3.4*
Defining message references of Message-driven Beans		Map the actual Connection Factory and Destination wherever the Connection Factory and Destination of Message-driven Bean are being referenced.	O	*9.4*
Defining transaction attributes		Define the transaction attributes.	O	*9.5*
Defining CMP of an Entity Bean		Define the CMP for delegating the persistence management of Entity Bean to a container.	O	*9.6*
	CMP settings	Define the settings for delegating the persistence management of an Entity Bean to a container, by using a single primary key. Define the settings for delegating the persistence management of an Entity Bean to a container, by using the compound primary key.	O	*9.6.1*
	Mapping CMP1.x and the database	Map the CMP1.x Entity Bean field to a table in the database.	O	*9.6.2*
	Mapping CMP2.x and the database	Map the CMP2.x Entity Bean field to a table in the database.	O	*9.6.3*
Defining the runtime attributes of Enterprise Beans		Set the runtime operations of the Enterprise Beans. Specify the settings when required.	Y	*9.10*
	Setting the runtime properties for Stateful Session Beans	Set the properties for runtime operations of Stateful Session Beans.	Y	*9.10.1*
	Setting the runtime properties for Stateless Session Beans	Set the properties for runtime operations of Stateless Session Beans.	Y	*9.10.2*

Items to be edited		Contents	Mandatory	Reference
	Setting the runtime properties for Entity Beans	Set the properties for runtime operations of Entity Beans.	Y	*9.10.3*
	Setting the runtime properties for Message-driven Beans	Set the properties for runtime operations of Message-driven Beans.	Y	*9.10.4*
Interceptor settings		Define the settings for using the interceptor.	O	*9.17*

Legend:

Y: Mandatory. Ensure that the settings are specified when customizing the J2EE applications that contain the corresponding Enterprise Beans.

O: Optional. Specify the settings based on the J2EE applications.

9.1.3 Property settings for servlets and JSPs

When you create a J2EE application containing the servlets and JSPs, specify the settings when required.

The following table describes the property settings of servlets and JSPs:

Table 9–2: Items to be edited in servlet and JSP properties

Items to be edited		Contents	Mandatory	Reference
Defining servlet and JSP references		Define the references (resource references) to Enterprise Beans, security role, and the resources (database or mail server).	O	*9.7*
	Defining Enterprise Bean references	Define the references to the Enterprise Beans.	O	*9.7.1*
	Defining mail configuration references	Define the references to the mail servers.	O	*9.7.2*
	Defining resource adapter references	Define the references to the resource adapters.	O	*9.7.3*
	Defining resource environment references	Define the references to the resource environment.	O	*9.7.4*
Defining servlet and JSP mapping		Define the mapping of servlets and JSPs.	O	*9.8*
Filter settings		Add the filter and define the mapping.	O	*9.9*
	Adding a filter	Add the filter.	O	*9.9.1*
	Defining a filter mapping	Define the mapping to the filter.	O	*9.9.2*
	Deleting a filter	Delete the filter.	O	*9.9.3*
Defining the runtime attributes of servlets and JSPs		Define the runtime attributes of servlets and JSPs.	O	*9.11*

Items to be edited		Contents	Mandatory	Reference
	Defining the context root of a J2EE application	Set the context root of the J2EE applications.	Y	*9.11.1*
	Defining the control of the number of threads executed concurrently for each Web application	In the Web container, specify whether to control the number of threads executed concurrently for each Web application and the thread count.	O	*9.11.2*
	Defining the control of the number of threads executed concurrently for each URL group	For each URL group, specify the definition for controlling the number of threads executed concurrently.	O	*9.11.3*
	Defining and monitoring the number of requests pending for each URL group	For each URL group, specify the definition for monitoring the number of pending requests.	O	*9.11.4*
Setting the default character encoding		Set the default character encoding for Web applications.	O	*9.12*
Error notification settings for servlets and JSPs		Specify the settings for reporting an error that occurs in the servlets or JSPs to the J2EE applications.	O	*9.16*

Legend:

Y: Mandatory. Ensure that the settings are specified when customizing the J2EE applications that contain servlets and JSPs.

O: Optional. Specify the settings based on the J2EE applications.

9.1.4 Property settings for J2EE applications (common)

Specify the following settings when required, regardless of the configuration of the J2EE applications:

Table 9–3: Items to be edited in J2EE application (common) properties

Items to be edited		Contents	Manda tory	Editing reference location for specific properties file	
				Reference manual#	Reference location
Referencing and changing the names registered in JNDI namespace		Reference a name registered in the JNDI namespace of the J2EE application and give an optional name if required.	O	This manual	*9.13*
	Referencing the J2EE application names	Specify the settings to reference a J2EE application name.	O	This manual	*9.13.1*
	Referencing and changing the Enterprise Bean name	Specify the settings to reference and change an Enterprise Bean name.	O	This manual	*9.13.2*

Items to be edited		Contents	Mandatory	Editing reference location for specific properties file	
				Reference manual#	Reference location
CTM scheduling		Specify the settings to use CTM for queue scheduling, and the scheduling method.	O	This manual	*9.14*
	Scheduling the J2EE applications	Specify the settings related to linking each J2EE application with CTM.	O	This manual	*9.14.1*
	Scheduling the Stateless Session Beans	Specify the settings for scheduling Stateless Session Beans.	O	This manual	*9.14.2*
Setting invocation order		Set the order for invoking the J2EE applications and the order for invoking an Enterprise Bean (EJB-JAR) and servlets or JSPs (WAR) included in the J2EE applications.	O	This manual	*9.15*
	Setting the invocation order for J2EE applications	Set the order for invoking the J2EE applications.	O	This manual	*9.15.1*
	Setting the invocation order for Enterprise Beans	Set the order for invoking the Enterprise Beans (EJB-JAR) included in the J2EE application.	O	This manual	*9.15.2*
	Setting the invocation order for servlets and JSPs	Set the order for invoking servlets and JSPs (WAR) included in the J2EE applications.	O	This manual	*9.15.3*
Setting security roles		This setting is necessary for performing user management with security roles. You can register and map the users and roles.	O	*Security Management Guide*	9.2
	Setting users	Register a user. You can also map the user to a role.	O	*Security Management Guide*	9.2.1
	Setting roles	Register a role. You can also map the role to a user.	O	*Security Management Guide*	9.2.2
Defining security role references		Define the references to the security role.	O	*Security Management Guide*	9.3
	Defining security role references of Enterprise Beans	Map the security role that is actually managed by the J2EE server, wherever the security role is referenced.	O	*Security Management Guide*	9.3.1
	Defining security role references of servlets and JSPs	Define the references to the security role.	O	*Security Management Guide*	9.3.2
Defining security (method permission)		Define the method permission for setting the access permission based on the security role. Define the method permission for setting the access permission for all users.	O	*Security Management Guide*	9.4
Defining security (security identity)		Set UseCallerIdentity as the security identity that forms the security information. Set Run as as the security identity that forms the security information.	O	*Security Management Guide*	9.5
	Security identity of the Enterprise Beans	Map the user ID that is actually managed by the J2EE server, to the security identity information referenced in the J2EE application.	O	*Security Management Guide*	9.5.1

Items to be edited		Contents	Mandatory	Editing reference location for specific properties file	
				Reference manual#	Reference location
	Security identity of servlets and JSPs	Define the security identity.	O	*Security Management Guide*	*9.5.2*

Legend:

O: Optional. Specify the settings based on the J2EE applications.

#

uCosminexus Application Server has been omitted from the manual names described in the *Reference manual* column.

9.2 Property settings using the HITACHI Application Integrated Property File

9.2.1 Setting procedure

You can acquire the HITACHI Application Integrated Property File by using the server management commands and collectively edit the contents of specific attribute files.

The specific attribute file elements included in the HITACHI Application Integrated Property File are described below:

- Application attribute file
  (*hitachi-application-property*/*hitachi-application-property*)
- EJB-JAR attribute file
  (*hitachi-ejb-jar-property*/*hitachi-ejb-jar-property*)
- Session Bean attribute file
  hitachi-session-bean-property/*hitachi-session-bean-property*
- Entity Bean attribute file
  hitachi-entity-bean-property/*hitachi-entity-bean-property*
- Message-driven Bean attribute file
  hitachi-message-bean-property/*hitachi-message-bean-property*
- WAR attribute file
  hitachi-war-property/*hitachi-war-property*
- Filter attribute file
  hitachi-filter-property/*hitachi-filter-property*
- Servlet attribute file
  hitachi-servlet-property/*hitachi-servlet-property*
- HITACHI Connector Property File
  hitachi-connector-property/*hitachi-connector-property*

For details on the HITACHI Application Integrated Property File, see *3.1 HITACHI Application Integrated Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

The following sections describe the procedures for setting the J2EE application properties by using the HITACHI Application Integrated Property File:

(1) Acquiring the HITACHI Application Integrated Property File

To acquire the HITACHI Application Integrated Property File, specify the path of the HITACHI Application Integrated Property File and execute the `cjgetappprop` command.

Execution Format

```
cjgetappprop server-name -name J2EE application-name -type all -c path-of
-the-HITACHI Application Integrated Property File
```

Example of execution

```
cjgetappprop Myserver -name App1 -type all -c App1prop.xml
```

(2) Editing the property settings

Edit the HITACHI Application Integrated Property File that is output in (1) above with a text editor. For the items to be edited in the J2EE application properties, see *9.1 Overview of the property settings of J2EE applications*. The items edited in the HITACHI Application Integrated Property File are the same as the items edited in the specific attribute files.

(3) Setting the application attributes

To apply the items specified in the HITACHI Application Integrated Property File to the application attributes, execute the `cjsetappprop` command.

Execution Format

```
cjsetappprop server-name -name J2EE application-name -type all -c path-of
-the-HITACHI Application Integrated Property File
```

Example of execution

```
cjsetappprop MyServer -name App1 -type all -c App1Prop.xml
```

9.3 Defining Enterprise Bean references

You define the references (resource references) of the Enterprise Beans that form a J2EE application. The types of Enterprise Beans are as follows:

- Session Bean
- Entity Bean
- Message-driven Bean

Each Enterprise Bean has the following three types of references:

- Enterprise Bean reference
  The Enterprise Bean reference is a reference for invoking other Enterprise Beans.
- Resource reference
  The resource reference is a reference to the resources. The resource references include the mail references, resource adapter references, and the resource environment references.
- Security role reference
  The security role reference is a role name reference that is used for executing access control for invoking the Enterprise Bean methods.

The following sections describe the Enterprise Bean references and the resource references. For the security role references, see *9.3 Definition of security role references* in the *uCosminexus Application Server Security Management Guide*.

For property settings, reference the following subsections:

- *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.6.1 Specifications of the HITACHI MessageDrivenBean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*

9.3.1 Defining other Enterprise Bean references

If the Enterprise Beans that form a J2EE application invoke other Enterprise Beans, set the properties for resolving the references.

(1) Attribute file to be edited

Edit the attribute file that maps to the types of Enterprise Beans forming the J2EE application.

- Session Bean attribute file
- Entity Bean attribute file
- Message-driven Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file

  Execute the following command to acquire an Enterprise Bean attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c
  C:\home\adder_ejb.xml
  ```

- Setting the attributes

  Execute the following command to apply the values of the Enterprise Bean attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute-file
  ```

  Example of execution

  ```
  cjsetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c
  C:\home\adder_ejb.xml
  ```

(3) Attribute settings to be edited

The access types of Enterprise Beans to be referenced are as follows:

Access types of Enterprise Beans to be referenced	Corresponding tags
Remote interface	<ejb-ref>
Local interface	<ejb-local-ref>

Note:

The business interface (*business-local* and *business-remote*) cannot be changed as it is set in annotation.

The individual settings are explained below:

(a) Defining Enterprise Bean references for a remote interface

The following table describes the reference settings (<ejb-ref>) for the remote Enterprise Beans:

Items	Mandatory	Corresponding tags
Description	O	<description>
Enterprise Bean reference name	Y	<ejb-ref-name>
Type of Enterprise Bean to be referenced	Y	<ejb-ref-type>
Remote home interface	Y	<home>
Remote interface	Y	<remote>

Items	Mandatory	Corresponding tags
<ejb-name> to be linked	O	<ejb-link>

Legend:
  Y: Mandatory
  O: Optional

(b) Defining the Enterprise Bean references for a local interface

The following table describes the reference settings (<ejb-local-ref >) for the local Enterprise Beans:

Items	Mandatory	Corresponding tags
Description	O	<description>
Enterprise Bean reference name	Y	<ejb-ref-name>
Type of Enterprise Bean to be referenced	Y	<ejb-ref-type>
Local home interface	Y	<local-home>
Local interface	Y	<local>
<ejb-name> to be linked	O	<ejb-link>

Legend:
  Y: Mandatory
  O: Optional

(4) Notes

- When invoking the Enterprise Bean of a remote interface, you can set the Enterprise Bean in '<ejb-name> to be linked' (<ejb-link>), considering the Enterprise Bean existing in another application as the mapping destination.

  If the Enterprise Bean to be mapped to is running on Cosminexus, you can specify the mapping destination in the following format by using the naming service switching functionality.

  Specification format

  ```
  corbaname::name-space-host-name:name-space-port-number#JNDI-name-of-the-EJ
  BHome-object-reference
  ```

  Name-space-host-name

  Specify the name of the host on which the namespace used by the mapping destination Enterprise Bean operates.

  Name-space-port-number

  Specify the port number on which the namespace used by the mapping destination Enterprise Bean operates.

  JNDI-name-of-the-EJBHome-object-reference

  Specify the JNDI name in the following format:

  ```
  HITACHI_EJB/SERVERS/server-name/EJB/J2EE-APP-name/Enterprise-Bean-name
  ```

  Server name: Name of a J2EE server

  J2EE-APP-name: Lookup name of a J2EE application

  Enterprise-Bean-name: Lookup name of an Enterprise Bean

  Example of specification

  ```
  corbaname::MyHost:900#HITACHI_EJB/SERVERS/MyServer/EJB/MyApplication/MyBean
  ```

- When a naming service operates on the local host, do not use the string 'localhost' in ejbserver.naming.host key of the `usrconf.properties` file for the J2EE server; instead, specify the machine name or the IP address.

9.3.2 Defining mail configuration references

If the Enterprise Beans forming a J2EE application reference a mail configuration, set the properties for defining the references.

(1) Attribute file to be edited

For details on the attribute file to be edited, see *(1) Attribute file to be edited* in *9.3.1 Defining other Enterprise Bean references*.

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes* in *9.3.1 Defining other Enterprise Bean references*.

(3) Attribute settings to be edited

The following table describes the reference settings (<resource-ref>) of a mail configuration:

Items	Mandatory	Corresponding tags
Description	O	<description>
Resource reference name	Y	<res-ref-name>
Resource type	Y	<res-type>
Resource authentication method	Y	<res-auth>
Scope for resource sharing	O	<res-sharing-scope>
Mail configuration name to be linked to	O	<linked-to>

Legend:
Y: Mandatory
O: Optional

9.3.3 Defining resource adapter references

If the Enterprise Beans forming a J2EE application reference a resource adapter, set the properties for resolving the references.

(1) Attribute file to be edited

For details on the attribute file to be edited, see *(1) Attribute file to be edited* in *9.3.1 Defining other Enterprise Bean references*.

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes* in *9.3.1 Defining other Enterprise Bean references*.

(3) Attribute settings to be edited

The following table describes the reference settings (<resource-ref>) of a resource adapter:

Items	Mandatory	Corresponding tags
Description	O	<description>
Resource reference name	Y	<res-ref-name>
Resource type	Y	<res-type>
Resource authentication method	Y	<res-auth>
Scope for resource sharing	O	<res-sharing-scope>
Resource adapter display name to be linked to	O	<linked-to>

Legend:

Y: Mandatory

O: Optional

9.3.4 Defining resource environment references

You set the properties for resolving the resource environment references of the Enterprise Beans forming a J2EE application.

(1) Attribute file to be edited

For details on the attribute file to be edited, see *(1) Attribute file to be edited* in *9.3.1 Defining other Enterprise Bean references*.

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes* in *9.3.1 Defining other Enterprise Bean references*.

(3) Attribute settings to be edited

The following table describes the reference settings (<resource-env-ref>) of the resource environment:

Items		Mandatory	Corresponding tags
Description		O	<description>
Resource environment variable name		Y	<resource-env-ref-name>
Type of the variable value of resource environment		Y	<resource-env-ref-type>
Linked queue information		O	<linked-queue>
	Display name of resource adapter	O	<resource-adapter>

Items		Mandatory	Corresponding tags
	Queue name	O	<queue>
JavaBeans resource display name to be linked to		O	<linked-to>

Legend:

Y: Mandatory

O: Optional

9.4 Defining message references of Message-driven Beans

9.4.1 Definition method

You define the message references of Message-driven Beans.

(1) Attribute file to be edited

Message-driven Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a Message-driven Bean attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation name -type ejb -resname EJB-JAR display name/Message-driven-Bean
-display-name -c path-of-the-Message-driven-Bean-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name message -type ejb -resname message/MyMessag
e -c C:\home\MyMessageBean.xml
```

- Setting the attributes
  Execute the following command to apply the values of the Message-driven Bean attribute files:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type ejb -resname EJB-JAR-display-name/Message-driven-Bean
-display-name -c path-of-the-Message-driven-Bean-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name message -type ejb -resname message/MyMessag
e -c C:\home\MyMessageBean.xml
```

(3) Attribute settings to be edited

The following table describes the message reference settings (<message-ref>):

Items	Mandatory	Corresponding tags
Connection Factory name	Y	<connection-factory>
Display name of the resource adapter	Y	<connection-destination> - <resource-adapter>
Display name of the queue#	Y	<connection-destination> - <queue>

Legend:
Y: Mandatory

#

We recommend that you do not share the JMS queue between several different Message-driven Beans.

For details on the property settings, see *3.6.1 Specifications of the HITACHI MessageDrivenBean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.5 Defining transaction attributes

9.5.1 Definition method

You define the method of managing transactions by using a container.

You can specify the transaction attributes for each Enterprise Bean, interface, and method. When the transaction attributes are not specified, the attributes specified for the upper level become applicable.

- Application in Enterprise Beans
  Apply the assigned transaction attributes to the methods in the Enterprise Bean.
- Application in interfaces
  Apply the assigned transaction attributes to the methods in the interface.
- Application in methods
  Apply the assigned transaction attributes to one method.

(1) Attribute file to be edited

From the following attribute files, edit the attribute file that maps to the type of Enterprise Bean that sets the transaction attributes:

- Session Bean attribute file
- Entity Bean attribute file
- Message-driven Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an Enterprise Bean attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c
  C:\home\adder_ejb.xml
  ```

- Setting the attributes
  Execute the following command to apply the values of the Enterprise Bean attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute-file
  ```

Example of execution

```
cjgetappprop MyServer -name adder -type ejb -resname adder/adder_eb -c
C:\home\adder_ejb.xml
```

(3) Attribute settings to be edited

The following table describes the settings for the container transaction attributes (<container-transaction>):

Items	Mandatory	Corresponding tags
Description	O	<description>
Method description	O	<method> - <description>
Interface classification	O	<method> - <method-intf>
Method name	Y	<method> - <method-name>
Transaction attribute	Y	<trans-attribute>

Legend:

Y: Mandatory

O: Optional

For property settings, reference the following subsections:

- *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.6.1 Specifications of the HITACHI MessageDrivenBean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*

The following points specify the transaction attributes and explain the transaction management operations:

- *NotSupported*
  The method is not executed in a container-managed transaction.
- *Supports*
  The process of method transaction differs on a case-to-case basis. Use this attribute only for the method that runs correctly inside the transaction and outside the transaction.
- *Required*
  The container always executes the method inside the transaction.
- *RequiresNew*
  The method is always executed in a new transaction.
- *Mandatory*
  The method must always use a client transaction. In other words, the client must invoke the method inside the transaction.
- *Never*
  The method cannot be executed inside the transaction. In other words, the client must invoke the method outside of the transaction.

(4) Notes

- The transaction attributes that you can set differ based on the type of Enterprise Bean. For details on the transaction attributes that you can set and the transaction management operations, see the *uCosminexus Application Server Common Container Functionality Guide*.
- In the Session Bean property (<transaction-type>), if it is specified that the Session Bean will manage its own transaction, then in this setting, the transaction attributes cannot be changed.
- In the case of the Message-driven Bean, you can only set the `onMessage` method.

9.6 Defining CMP of an Entity Bean

You specify the CMP definition of an Entity Bean. If CMP is defined, you can delegate the persistence management of Entity Bean that is an Enterprise Bean, to the container.

9.6.1 CMP settings

(1) Attribute file to be edited

Entity Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file

  Execute the following command to acquire an Entity Bean attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URLL] -name J2EE appli
  cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
  -name -c <path-of-the-Entity-Bean-attribute-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name adder -type ejb -resname account/MyAccoub
  -c C:\home\adder_ejb.xml
  ```

- Setting the attributes

  Execute the following command to apply the values of the Entity Bean attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
  cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
  -name-c path-of-the-Entity-Bean-attribute-file
  ```

  Example of execution

  ```
  cjsetappprop MyServer -name adder -type ejb -resname account/MyAccoub
  -c C:\home\adder_ejb.xml
  ```

(3) Attribute settings to be edited

The method of defining CMP differs based on whether the primary key is a single primary key or a compound primary key.

(a) Setting the attributes for a single primary key

You set the properties for a single primary key.

Check the persistence management type (<persistence-type>). In the case of CMP Entity Bean, 'Container' is set.

The following table describes the property settings for a single primary key:

Items	Mandatory	Corresponding tags
Primary key class[#1]	Y	<prim-key-class>
Scope of reentrant	Y	<reentrant>
Description of the persistence management field	O	<cmp-field> - <description>
Field name of the persistence management field[#2]	O	<cmp-field> - <field-name>
Field name of the primary key field	O	<primkey-field>

Legend:

Y: Mandatory

O: Optional

#1

Enter a class or interface that contains the primary key of this Entity Bean. The primary key class must be either a java.lang.Object class or a class or an interface in the same container-managed field.

#2

Normally, the field name is not required. If you want to add a field name, the field name also needs to be added in the class definition of the Entity Bean beforehand.

For details on the property settings, see *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Setting the attributes for a compound primary key

You set the properties for a compound primary key.

Check the persistence management type (<persistence-type>). In the case of a CMP Entity Bean, 'Container' is set.

The following table describes the property settings for a compound primary key:

Items	Mandatory	Corresponding tags
Primary key class[#1]	Y	<prim-key-class>
Scope of reentrant	Y	<reentrant>
Description of the persistence management field	O	<cmp-field> - <description>
Field name of the persistence management field[#2]	O	<cmp-field> - <field-name>

Legend:

Y: Mandatory

O: Optional

#1

Enter a class or an interface that contains the primary key of this Entity Bean. The primary key class must be either a java.lang.Object class or a class or an interface in the same container-managed field.

#2

Normally, the field name is not required.

If you want to add a field name, the field name also needs to be added in the class definition of the Entity Bean beforehand.

For details on the property settings, see *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.6.2 Mapping CMP1.x and the database

This section explains the mapping of the fields of a CMP1.x Entity Bean to a table in the database.

(1) Attribute file to be edited

Entity Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an Entity Bean attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
-name-c path-of-the-Entity-Bean-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type ejb -resname account/MyAccoub
-c C:\home\adder_ejb.xml
```

- Setting the attributes
  Execute the following command to apply the values of the Entity Bean attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
-name-c path-of-the-Entity-Bean-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type ejb -resname account/MyAccoub
-c C:\home\adder_ejb.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for mapping the fields of CMP1.x Entity Bean to a table in the database (<cmp-map>):

Items	Mandatory	Corresponding tags
Display name of the resource adapter	Y	<datasource-name>
Catalog name of the database	O	<catalog-name>
Schema name of the database	O	<schema-name>
Table name of the database	Y	<table-name>
Permission or prohibition of writing to the database	Y	<read-only-access>
Transaction isolation level[#1]	O	<transaction-isolation>
Method of comparing the data by writing to the database	O	<concurrency-protection>
Information of mapping between the field and the table column[#2]	Y	<field-impl>

Items	Mandatory	Corresponding tags
Search conditions of the `finder` method[#3]	Y	<finder-impl>

Legend:

Y: Mandatory

O: Optional

#1

The value of the transaction isolation level (<transaction-isolation>) that can be used differs based on the options supported in the database and the JDBC driver.

#2

Set a column of the database table for the primary key. This column is also used for mapping the fields of the same database table.

The value of the field name of the EntityBean (<field-name>) cannot be changed. Set a database column (<column-name>) for mapping the field name of the Entity Bean (<field-name>).

The information about the mapping between the fields and the table column (<field-impl>) consists of the following:

Items	Corresponding tags
Field name of the EntityBean	<field-name>
Column name of the table	<column-name>

#3

The value of the method name (<method-name>) of the `finder` method cannot be changed. Set the search condition (<where-clause>) for the method name table (<method-name>) in the `finder` method.

The `finder` method information in the Enterprise Bean (<finder-impl>) consists of the following:

Items	Corresponding tags
Method name of the `finder` method	<method-name>
Search condition for the table	<where-clause>

For details on the property settings, see *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(4) Notes

- The function for mapping the CMP field of an Entity Bean to multiple tables in the database is not provided.
- If the user specified in the DB Connector definition to be used for mapping and the owner of the database table that forms the CMP mapping destination, are different, the user specified in the DB Connector definition needs the following permissions:
  For mapping: SELECT
  For execution: SELECT, INSERT, UPDATE, DELETE

9.6.3 Mapping CMP2.x and the database

You map the fields of CMP2.x Entity Bean to a database table.

This section explains how to define the relationship (*CMR*) between two CMP2.x Entity Beans that define CMP.

The procedure of mapping CMP2.x and the database is as follows:

1. If there is a CMR relationship between the CMP2.x Entity Beans, define CMR between the CMP2.x Entity Beans.
2. Map the fields of the CMP2.x Entity Bean to a table in the database.
3. Execute the `cjgencmpsql` command for the CMP2.x Entity Beans and generate an SQL statement.

(1) Defining CMR

If there is a CMR relationship between the CMP2.x Entity Beans that define CMP, define the CMR between the CMP2.x Entity Beans.

(a) Attribute file to be edited

EJB-JAR attribute file

(b) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an EJB-JAR attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name -c path-of-the-EJB-J
  AR-attribute-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder -c C:\home\a
  dder_ejb.xml
  ```

- Setting the attributes
  Execute the following command to apply the values of the EJB-JAR attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name -c path-of-EJB-JAR-a
  ttribute-file
  ```

  Example of execution

  ```
  cjsetappprop MyServer -name adder -type ejb -resname adder -c C:\home\a
  dder_ejb.xml
  ```

Note:

You can acquire the EJB-JAR attribute file that is included in a running application, but you cannot apply the relation information that is defined.

(c) Attribute settings to be edited

The following table describes the property settings that define the relationship between two CMP2.x Entity Beans (<relationships> - <ejb-relation>):

Items	Mandatory	Corresponding tags
Description	O	<description>
Relation definition name	Y	<ejb-relation-name>
Enterprise Bean(EJB1) information	Y	<ejb1>
Enterprise Bean(EJB2) information	Y	<ejb2>

Legend:

Y: Mandatory

O: Optional

For details on the property settings, see *3.3.1 Specifications of the HITACHI EJB-JAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For the Enterprise Bean (EJB1) information (<ejb1>) and Enterprise Bean (EJB2) information (<ejb2>), specify the items for the relationship between the Enterprise Bean on the setting side and the Enterprise Bean on the other side respectively.

Items	Mandatory	Corresponding tags
Description	O	<description>
Role name of relationship	O	<ejb-relationship-role-name>
Multiplicity of setting side	Y	<multiplicity>
Setting of cascade delete	Y	<cascade-delete>
<ejb-name> of setting side	Y	<ejb-name>
Name of CMR field	Y	<cmr-field-name>
Type of CMR field	O	<cmr-field-type>

Legend:

Y: Mandatory

O: Optional

For details on the property settings, see *3.3.1 Specifications of the HITACHI EJB-JAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(2) Mapping CMP2.x and the database

You map the fields of a CMP2.x Entity Bean to a table in the database.

(a) Attribute file to be edited

For the attribute file to be edited, see *(1) Attribute file to be edited* in *9.6.2 Mapping CMP1.x and the database*.

(b) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes* in *9.6.2 Mapping CMP1.x and the database*.

(c) Attribute settings to be edited

The following table describes the property settings for mapping the fields of a CMP2.x Entity Bean with a table in the database (<cmp-map>):

Items	Mandatory	Corresponding tags
Display name of the resource adapter	Y	<datasource-name>
Catalog name of the database	O	<catalog-name>
Schema name of the database	O	<schema-name>
Table name of the database	Y	<table-name>
Permission or prohibition of writing to the database	Y	<read-only-access>
Transaction isolation level#1	O	<transaction-isolation>
Method of comparing the data by writing to the database	O	<concurrency-protection>
Information of mapping between the field and the table column#2	Y	<field-impl>

Legend:

Y: Mandatory

O: Optional

#1

The value of the transaction isolation level (<transaction-isolation>) that can be used differs based on the option supported in the database and the JDBC driver.

#2

Set a column of the database table for the primary key. This column is also used for mapping the fields of the same database table.

The value of the field name of an EntityBean (<field-name>) cannot be changed. Set a database column (<column-name>) for mapping the field name of the Entity Bean (<field-name>).

The information about mapping between the fields and the table column (<field-impl>) consists of the following items:

Items	Corresponding tags
Field name of the EntityBean	<field-name>
Column name of the table	<column-name>

For details on the property settings, see *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(d) Notes

- The function for mapping the CMP field of an Entity Bean to multiple tables in the database is not provided.
- When the user specified in the DB Connector definition to be used for mapping and the owner of the database table that forms the CMP mapping destination, are different, the user specified in the DB Connector definition needs the following permissions:

  For mapping: SELECT

  For execution: SELECT, INSERT, UPDATE, DELETE
- When the mapping of the CMP fields is not complete and the application is running, an SQL statement cannot be generated.
- A table of two Beans forming a CMR relationship must be in the database of the same database product.

(3) Generating an SQL statement

You generate the SQL statement that is used when the `finder` or `select` method is executed.

When there is no CMR relationship between CMP2.x Entity Beans

Generate an SQL statement on the basis of the defined EJB QL (<query>).

When there is a CMT relationship between CMP2.x Entity Beans

Before generating an SQL statement, map the fields of all the CMP2.x Entity Beans to the columns of the table. After the mapping for all the Beans is complete, generate the SQL statements for all the CMP2.x Entity Beans. Generate the SQL statement similarly, even if the `finder` or `select` method does not exist in the Entity Bean.

Note that when there is a CMR relationship between the Entity Beans, in addition to the SQL statements used for executing the `finder` or `select` method, the SQL statements for operating the tables for CMR are also generated. If the CMR settings are changed after the SQL statement is generated, generate an SQL statement for the Entity Bean related to the changed CMR. For details on the tables for CMR, see the section *(4) Tables for CMR*.

Execute the following command to generate an SQL statement:

- For generating the SQL statements of all the CMP2.x Entity Beans included in a J2EE application

Execution format

```
cjgencmpsql [server-name] -name J2EE application name
```

Example of execution

```
cjgencmpsql MyServer -name App1
```

- For generating the SQL statements of a specific CMP2.x Entity Bean included in a J2EE application

Execution format

```
cjgencmpsql [server-name] -name J2EE application name -resname EJB-JAR-dis
play-name/ Entity-Bean-display-name
```

Example of execution

```
cjgencmpsql MyServer -name App1 -resname EjbJar1/Ejb1
```

For details on the `cjgencmpsql` command, see *cjgencmpsql (generate SQL statements for CMP2.x Entity Beans)* in the *uCosminexus Application Server Command Reference Guide*.

(4) Tables for CMR

This section describes the tables for CMR.

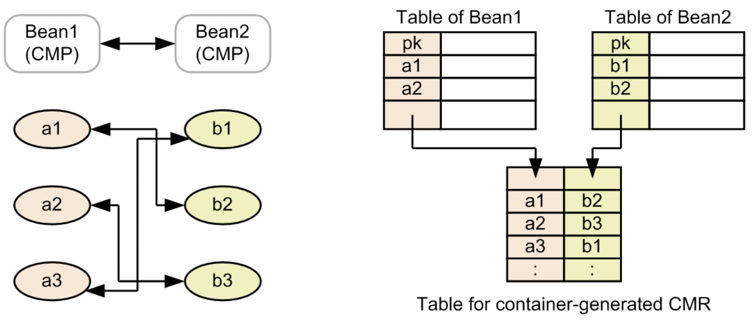
Note that in the explanation hereafter, the Entity Bean set in <ejb1> - <ejb-name> of EJB-JAR is called *Bean1* and the Entity Bean set in <ejb2> - <ejb-name> is called *Bean2*.

(a) Overview of tables for CMR

In the EJB container, if there is a CMR relationship between the CMP2.x Entity Beans, a table is created in the database and it is used for maintaining the CMR status.

As shown in the following figure, the primary keys of two Entity Beans that are related are stored in the table for CMR:

Figure 9–1: Table for CMR (in the case of a one-to-one and a two-way relationship)



(b) Generating and deleting the tables for CMR

If an application that contains CMR is started, a table is generated for CMR. The table is generated in the schema that has the Bean1 table. This table is deleted when the application is stopped. For creating, deleting, and operating the tables, you use the SQL statement that is generated after database mapping when the application is customized.

If the server is stopped while the application is running, the table for CMR remains in the database. If the server is restarted while the application is running, the existence of the table for CMR in the database is checked. The following table describes the process of generating the table for CMR when an application and server are started:

Table 9–4: Process for generating the table for CMR when an application and server are started

Status	Does a table with the same name as that of the table to be created exist in the database?	If a table with the same name exists, are the number of columns, name, and the JDBC type the same?	Operations	Message ID that is output
When starting[#1]	No	Not applicable	Generate and use the table.	KDJE43007-I
	Yes	Same or not the same	Startup is suspended.[#2]	KDJE43003-E
When starting the J2EE server	No	Not applicable	Startup is suspended.[#3]	KDJE43008-E
	Yes	Same	When starting, assume that the table has been created for CMR and use it.	KDJE43006-I
		Not the same	Startup is suspended.[#3]	KDJE43008-E

#1

This is in case of default, especially when the option, at the start of the application that contains CRM, is not set. For details on starting options, see *(e) Restoring the table for CMR when a failure occurs*.

#2

The application fails to start if a table with the same name already exists in the database. If the existing table is not required, either delete the table manually by using the tool provided in the database products or re-execute the process of generating the SQL statement. If you choose to re-execute the process of generating the SQL statement, the table is given a name that is not repeated in the database schema, and the SQL statement is generated again (Generate the SQL statements for all the methods of all the EJBs in the EJB-JAR. If this process is not executed, the change in the table name may not be applied to all the SQL statements). After changing the table name, restart the application.

#3

The J2EE server is started when the target application is not running.

(c) Naming conventions for the tables for CMR

The name of the table for CMR is as follows:

- *In the case of a two-way relationship*
  [<ejb-name> of Bean1]_ [<cmr-field-name> of Bean1]_[<ejb-name> of Bean2]_[<cmr-field-name> of Bean2]
- *In the case of a one-way relationship (one-way is considered to be from Bean1 to Bean2)*
  [<ejb-name> of Bean1]_[<cmr-field-name> of Bean1]_[<ejb-name> of Bean2]

If the name decided by this method is of 29 characters or more, it will be truncated to 28 characters. If the name that is truncated to 28 characters matches with a table name already existing in the database or if it overlaps with the name of the table for another CMR in the same EJB-JAR, then it is differentiated by adding a number from 0 to 99 at the end.

(d) Precautions related to CMR

- The user needs the permission to execute the `CREATE TABLE` command in the schema in which the Bean1 table exists.
- The generation of the table for CMR may fail due to the restriction on the size of the primary key depending on the type of the database and OS. The primary key of the Entity Bean is saved in the table for CMR, as shown in Figure 9-1. Adjust the size of the primary key of the Entity Bean.
- When the application is stopped, the table for CMR is deleted, hence, the information on the relation that was established until then, is lost. The relation information is also lost when the table for CMR is deleted manually.
- If the numbers from 0 to 99 are all used up for naming the table for CMR, the table for CMR cannot be generated. In this case, delete the tables in the database that are not required so that you can use those names, or see *(c) Naming conventions for the tables for CMR* for other names.
- Map the primary key of the relation-forming Entity Bean to a type that can be considered as the primary key of database table.
- When an application that contains CMR is stopped, the table for CMR is deleted, however, there are cases when the deletion of the table for CMR fails due to occurrence of a failure such as inability to access the database. In this case, the table remains in the database, so when it is not required, delete it manually by using the tools provided in the database products.

(e) Restoring the table for CMR when a failure occurs

When an application that contains CMR is started, if a failure occurs while the J2EE server was stopped for maintenance and then restarted, the J2EE server may not restart with the running application that contains CMR. After resolving the error, even if you try to restart the application that contains CMR, then, as described in Table 9-4, the application cannot be started if there is a table in the database with the same name as that of the table to be created (in order to avoid sharing of tables between the applications).

If the process of generating an SQL statement is executed again, an SQL statement that uses a new table name for CMR, is generated, and you can start the application by using the new table for CMR. If you want to continue the relation that was being used before stopping the J2EE server, you must start the application by using the table that is remaining in the database.

The ejbserver.ejb.cmp20.cmr.use.existing_table key in the `usrconf.properties` file is an option for restoring the relationship information that was being used until the start up of the application failed. If `true` is specified in this key, you can start the application by using the table that already exists in the database. If nothing is specified or if `false` is specified in this key, execute the operations described in Table 9-4. The procedure for using the existing table by using this option is as follows:

1. Use the tools provided in the database products to confirm that the table for CMR that was used before the failure is still in the database.
2. Stop the J2EE server (Take measures against the cause of failure in starting the application while it is running)

3. Set ejbserver.ejb.cmp20.cmr.use.existing_table=true in the `usrconf.properties` file.

4. Start the J2EE server.

5. Restart the application containing CMR that had failed to start.

   Note:

   Do not re-execute the process of SQL statement generation. If you re-execute the process, an SQL statement is generated with a new table name and you will not be able to use the earlier table.

6. Stop the J2EE server.

7. Set ejbserver.ejb.cmp20.cmr.use.existing_table=false in the `usrconf.properties` file or return to the state where this option is not set.

8. Restart the J2EE server.

9.7 Defining servlet and JSP references

You define the references (resource references) of the Web applications (servlets and JSPs) that form a J2EE application. The references are as follows:

- Enterprise Bean references
  This is the reference for invoking an Enterprise Bean.
- Resource references
  This is the reference for the database or mail server. The references include references to the mail, resource adapter, and resource environment.

9.7.1 Defining Enterprise Bean references

If the Web applications (servlets and JSPs) that form a J2EE application invoke an Enterprise Bean, set the properties for resolving the references.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a WAR attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder -c C:\home\a
dder_war.xml
```

- Setting the attributes
  Execute the following command to apply the values of the WAR attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder -c C:\home\a
dder_war.xml
```

(3) Attribute settings to be edited

The settings are similar to the property settings when an Enterprise Bean invokes another Enterprise Bean. See *(3) Attribute settings to be edited in 9.3.1 Defining other Enterprise Bean references*.

9.7.2 Defining mail configuration references

If the Web applications (servlets and JSPs) forming the J2EE application reference a mail configuration, set the properties for defining the references.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes in 9.7.1 Defining Enterprise Bean references*.

(3) Attribute settings to be edited

The settings are similar to the property settings when an Enterprise Bean references a mail configuration. See *(3) Attribute settings to be edited* in *9.3.2 Defining mail configuration references*.

9.7.3 Defining resource adapter references

If a Web application (servlets and JSP) forming a J2EE application references a resource adapter, set the properties for defining the references.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes in 9.7.1 Defining Enterprise Bean references*.

(3) Attribute settings to be edited

The settings are similar to the property settings when an Enterprise Bean references the resource adapter. See *(3) Attribute settings to be edited* in *9.3.3 Defining resource adapter references*.

9.7.4 Defining resource environment references

If a Web application (servlets and JSPs) forming a J2EE application references a resource environment, set the properties for defining the references.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

For details on acquiring the attribute file to be edited and setting the attributes, see *(2) Acquiring the attribute file to be edited and setting the attributes in 9.7.1 Defining Enterprise Bean references*.

(3) Attribute settings to be edited

The settings are similar to the property settings when an Enterprise Bean references a resource environment. See *(3) Attribute settings to be edited* in *9.3.4 Defining resource environment references*.

9.8 Defining servlet and JSP mapping

9.8.1 Definition method

You define the mapping of servlets and JSPs.

(1) Attribute file to be edited

Servlet attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a servlet attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type war -resname WAR-display-name/servlet-and-JSP-display
-name -c path-of-the-servlet-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder/adder_sv -c
C:\home\adder_war.xml
```

- Setting the attributes
  Execute the following command to apply the values of the servlet attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type war -resname WAR-display-name/servlet-and-JSP-display
-name -c path-of-the-servlet-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder/adder_sv -c
C:\home\adder_war.xml
```

(3) Attribute settings to be edited

Item	Mandatory	Corresponding tag
URL pattern	Y	<url-pattern>

Legend:
  Y: Mandatory

For details on the property settings, see *3.9.1 Specifications of the HITACHI Servlet Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.9 Filter settings

This section explains how to specify the filter settings. To set a filter, add the filter to a WAR file and define the mapping.

9.9.1 Adding a filter

Add a filter to a WAR file with the following procedures:

1. Create the filter attribute file.
   Create the filter attribute file by using a text editor. Specify the filter name and the class name of the filter in the filter attribute file. When required, specify the name and the value of the initialization parameter.
   For details on the filter property file, see *3.8 HITACHI Filter Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

2. Execute the following command to register the filter in the WAR file in a J2EE application:
   Execution format

```
cjaddapp [server-name] [-nameserver provider-URL] -type filter -name J2
EE application-name -warname WAR-display-name-for-adding-the-filter -c
path-of-the-filter-attribute-file
```

   Example of execution

```
cjaddapp MyServer -type filter -name App1 -warname account-war -c FilterPr
op.xml
```

   For details on the `cjaddapp` command, see *cjaddapp (add resource)* in the *uCosminexus Application Server Command Reference Guide*.

9.9.2 Defining a filter mapping

After adding the filter, you define the filter mapping.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a WAR attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

- Setting the attributes
  Execute the following command to apply the values of the WAR attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

(3) Attribute settings to be edited

The following table describes the property settings of the filter mapping definition (<filter-mapping>) of Web application (servlets and JSPs):

Items	Mandatory	Corresponding tags
Filter name	Y	<filter-name>
URL pattern	Y#2	<url-pattern>
Servlet name	Y#2	<servlet-name>
Filter application condition#1	O	<dispatcher>

Legend:
Y: Mandatory
O: Optional

#1
This attribute cannot be set in a WAR file of a version earlier than Servlet2.3.

#2
Specify either the URL pattern or the servlet name for filter mapping.

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.9.3 Deleting a filter

Execute the following command to delete the filter:

Execution format

```
cjdeleteapp [<server-name>] [-nameserver <provider-URL>] -name <J2EE appl
ication-name> -type filter -resname <Display-name-of-the-WAR-that-is-to-be
-deleted>/<Display-name-of-the-filter-that-is-to-be-deleted>
```

When deleting the filter for each WAR file, specify the display name of the WAR that is to be deleted. For deleting only the filter, specify the display names of the WAR file and the filter that are to be deleted.

Example of execution

```
cjdeleteapp MyServer -name App1 -type filter -resname account-war/account
-filter
```

For details on the `cjdeleteapp` command, see *cjdeleteapp (delete J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

9.10 Defining the runtime attributes of Enterprise Beans

You define the runtime attributes for the following Enterprise Beans:

- Stateful Session Bean
- Stateless Session Bean
- Entity Bean
- Message-driven Bean

9.10.1 Setting the runtime properties for Stateful Session Beans

Set the runtime properties of an application for the individual Stateful Session Beans that form the application.

(1) Attribute file to be edited

Session Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a Session Bean attribute file:

  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type ejb -resname EJB-JAR-display-name/Stateful-Session-Bea
n-display-name -c path-of-the-Session-Bean-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type ejb -resname adder/MyAdder -c C
:\home\MyAdder.xml
```

- Setting the attributes
  Execute the following command to apply the values of the Session Bean attribute file:

  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type ejb -resname EJB-JAR-display-name/Stateful-Session-Bea
n-display-name -c path-of-the-Session-Bean-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type ejb -resname adder/MyAdder -c C
:\home\MyAdder.xml
```

(3) Attribute settings to be edited

The runtime properties of the Stateful Session Bean are classified into the following two types:

- Runtime attributes common to the Session Beans

- Attributes specific to the Stateful Session Beans

The respective settings are described below:

(a) Runtime attributes common to the Session Beans

The following table describes the property settings of runtime attributes (<runtime>) common to the Session Bean:

Items	Mandatory	Corresponding tags
Look up name of the Enterprise Beans#1	Y	<lookup-name>
Optional name of the Enterprise Bean name having a remote interface#1	O	<optional-name>
Optional name of the Enterprise Bean name having a local interface#1	O	<local-optional-name>
Maximum number of sessions#2	O	<maximum-sessions>
Pass by reference settings#3	O	<pass-by-reference>

Legend:
Y: Mandatory
O: Optional

#1

For referencing and changing the name registered in the JNDI namespace, see *9.13.2 Referencing and changing the Enterprise Bean name*.

#2

The maximum number of sessions specified in the Stateful Session Bean is the maximum number of Stateful Session Beans that can be used from an EJB client application. If '0' is specified, the number of sessions (EJB objects) of the Stateful Session Beans that can be generated concurrently becomes infinite (the real maximum value is 2147483647).

#3

Pass by reference settings can be implemented in the `usrconf.properties` file also. If the settings are specified in either the properties or in the `usrconf.properties` file, the pass by reference settings become valid.

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Attributes specific to Stateful Session Beans

The following table describes the property settings of runtime attributes specific to the Stateful Session Bean (<runtime> - <stateful>):

Items	Mandatory	Corresponding tags
Maximum number of active sessions	Y	<maximum-active-sessions>
Period for which the bean is passivated, before it is reactivated	Y	<inactivity-timeout>
Period for which the bean is passivated, before the session is deleted	Y	<removal-timeout>

Legend:
Y: Mandatory

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

The property settings and operations are explained below:

Maximum number of active sessions

The generated Stateful Session Beans have the following two types of states:

- method-ready state
- passive state

The Stateful Session Beans in the method-ready state are already executable when accessed from an EJB client application.

The Stateful Session Bean in the passive state is activated and can be executed when it is in the method-ready state.

In the maximum number of active sessions, specify the maximum number of Stateful Session Beans that can concurrently be in a method-ready state. This value is the maximum number of Stateful Session Beans for which execution can start immediately, when accessed from multiple EJB client applications. If the number of Stateful Session Beans that are generated is more than the number specified here, some of the Stateful Session Beans that are in method-ready state are passivated. Among the Stateful Session Beans in the method-ready state, the Beans that are in a transaction are not passivated. If '0' is specified in the maximum number of sessions or the maximum number of active sessions, the passivation function does not work (The state of the Stateful Session Bean does not change from method-ready state to passive state).

Period for which the bean is passivated before it is reactivated (Default: 0 (minutes))

Specify the timeout value in minutes. If the Stateful Session Bean that is in a passive state, does not get activated until the lapse of the specified time period, this Bean is deleted. If '0' is specified in the period for which the Stateful Session Bean remains in a passive state until the bean is reactivated, then a timeout does not occur.
In the deletion of Stateful Session Bean instance, at maximum there may be a delay of the time specified in ejbserver.container.passivate.scan.interval, for the specified <inactivity-timeout> value. If you want to change this start up interval, specify the start up interval in the system property ejbserver.container.passivate.scan.interval of the `usrconf.properties` file in seconds. By default, 0 seconds are specified. An example of the specification of ejbserver.container.passivate.scan.interval is given below:

When you want to start the delete function of a Stateful Session Bean that is in a passive state, at 10-minute intervals

```
ejbserver.container.passivate.scan.interval=600
```

Period for which the bean is passivated before the session is deleted (Default: 0 (minutes))

Specify the timeout value in minutes. If the Stateful Session Bean that is in a method-ready state, is not used even once during the period that is specified here, this Bean is deleted. Among the Stateful Session Beans in the method-ready state, the Beans in a transaction are not deleted.

If '0' is specified in the period for which the Stateful Session Bean remains in a passive state until the session is deleted, then a timeout does not occur. In the deletion of Stateful Session Bean instance, at maximum there may be a delay of the time specified in ejbserver.container.remove.scan.interval, for the specified <removal-timeout> value. If you want to change this start up interval, specify the start up interval in ejbserver.container.passivate.scan.interval of `usrconf.properties` file in minutes. An interval of 5 minutes is assumed by default. An example of specification of ejbserver.container.remove.scan.interval is given below:

When you want to start the delete function of a Stateful Session Bean that is in a method-ready state, at 10-minute intervals

```
ejbserver.container.remove.scan.interval=10
```

(4) Notes

- The default user definition is 'a function that invokes `ejbActivate` and `ejbPassivate` and saves and restores the state of the Stateful Session Bean' and by default, the activation and passivation of the Stateful Session Bean does not work. As a result, the <maximum-active-sessions> setting that can be specified as the Enterprise Bean runtime property of the Stateful Session Bean, is not valid. In addition, the function for deleting the passivated Stateful Session Bean by timeout does not work. As a result, the <inactivity-timeout> setting that can be specified as the Enterprise Bean runtime property of the Stateful Session Bean is not valid.

- When using the activation and passivation functions, specify `true` for the ejbserver.stateful.passivate.switch key in the `usrconf.properties` file for the J2EE server.
- When the passivation function is running, the state of the Stateful Session Bean is saved by writing to a file and the instance is destroyed. On the other hand, when the activation function is running, the state of the Stateful Session Bean is read from the file and the instance is restored. When the activation and passivation functions are running, the objects set in the member variables of the Stateful Session Bean (or the classes referenced by the Bean) are also saved and restored. At this time, the following items can be saved and restored:
  - Objects that can be serialized
  - EJB object references
  - EJB home object references
  - javax.naming.Context obtained by "java: comp/env" lookup
  - javax.ejb.SessionContext

  Note that javax.jta.UserTransaction cannot be saved and restored, hence, do not save the object in the member variable. If the objects that cannot be saved or restored are stored in the member variable, release (substitute with null in the member variable) such objects using `ejbPassivate` and acquire the objects again by using `ejbActivate`.
- Specify the maximum number of sessions of the Stateful Session Bean that can be generated concurrently and the maximum number of Stateful Session Beans that are in the method-ready state, such that the following condition is fulfilled:

  ```
  Maximum number of Stateful Session Beans in the method ready state ≤ Maxim
  um number of sessions of the Stateful Session Beans that can be generated
  concurrently
  ```

  When you specify the maximum number of sessions of the Stateful Session Beans that can be generated concurrently, you also need to specify the maximum number of Stateful Session Beans that are in the method-ready state.

9.10.2 Setting the runtime properties for Stateless Session Beans

You set the application runtime properties for the individual Stateless Session Beans that form an application.

(1) Attribute file to be edited

Session Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a Session Bean attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Stateless-Session-Be
  an-display-name -c path-of-the-Session-Bean-attribute-file
  ```

Example of execution

```
cjgetappprop MyServer -name adder -type ejb -resname adder/MyAdder -c C
:\home\MyAdder.xml
```

- Setting the attributes

  Execute the following command to apply the values of the Session Bean attribute file:

  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type ejb -resname EJB-JAR-display-name/Stateless-Session-Be
an-display-name -c <path-of-the-Session-Bean-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type ejb -resname adder/MyAdder -c C
:\home\MyAdder.xml
```

(3) Attribute settings to be edited

The runtime properties of the Stateless Session Bean are classified into the following three types:

- Runtime attributes common to the Session Beans
- Attributes specific to the Stateless Session Bean
- Runtime attributes of CTM integration

The respective settings are described below:

(a) Runtime attributes common to the Session Beans

The following table describes the property settings for the runtime attributes common to the Session Beans (<runtime>):

Items	Mandatory	Corresponding tags
Look up name of the Enterprise Bean#1	Y	<lookup-name>
Optional name of the Enterprise Bean name having a remote interface#1	O	<optional-name>
Optional name of the Enterprise Bean name having a local interface#1	O	<local-optional-name>
Maximum number of sessions#2	O	<maximum-sessions>
Settings for scheduling#3	O	<enable-scheduling>
Pass by reference settings#4	O	<pass-by-reference>

Legend:

Y: Mandatory

O: Optional

#1

For referencing and changing the name registered in the JNDI namespace, see *9.13.2 Referencing and changing the Enterprise Bean name*.

#2

The maximum number of sessions specified in the Stateless Session Bean is the number of Stateless Session Bean instances that a client can use. This specified value, however, is not enabled.

#3

For details on CTM scheduling, see *9.14 CTM scheduling*.

#4

Pass by reference settings can also be specified in the `usrconf.properties` file. If the settings are specified in either the properties or in the `usrconf.properties` file, pass by reference settings become valid.

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Attributes specific to Stateless Session Bean

The following table describes the property settings for the runtime attributes specific to Stateless Session Beans (<runtime> - <stateless>):

Items	Mandatory	Corresponding tags
Minimum number of the instances in the pool	Y	<pooled-instance> - <minimum>
Maximum number of the instances in the pool	Y	<pooled-instance> - <maximum>

Legend:

Y: Mandatory

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

The property settings and operations are explained below:

Minimum number of instances in the pool

The number of Stateless Session Beans specified here are generated and pooled when the J2EE application starts. The number of pooled Stateless Session Beans lies between the maximum and the minimum count, depending on the volume of access from the EJB client applications. If '0' is specified in the minimum number of instances in the pool, the Stateless Session Bean is not generated when the J2EE application starts.

The minimum number also specifies the number of Stateless Session Bean instances created when the application is started.

Maximum number of instances in the pool

The generated Stateless Session Beans are pooled in the method-ready state. The pooled Stateless Session Beans are already executable when accessed from the EJB client applications. Specify the maximum number of pooled Stateless Session Beans in the maximum number of instances in the pool. The maximum number is the maximum number of the Stateless Session Beans that can be executed immediately when accessed from multiple EJB client applications. If '0' is specified in the maximum number of instances in the pool, the pooled Stateless Session Beans becomes infinite.

(c) Runtime attributes of CTM integration

The following table describes the property settings for the runtime attributes of CTM integration (<runtime> - <scheduling>).

Items	Mandatory	Corresponding tags
Queue name	Y	<queue-name>
Thread name	Y	<parallel-count>

Legend:

Y: Mandatory

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For details on CTM scheduling, see *9.14 CTM scheduling*.

(4) Notes

- The function for checking the upper limit of the Stateless Session Beans accessible from a client does not work. Instead set the value in *maximum number of instances in the pool.*
- Specify the minimum number of pooled Stateless Session Beans and the maximum number of pooled Stateless Session Beans such that the following condition is fulfilled:

  ```
  Minimum number of pooled Stateless Session Beans ≤ Maximum number of poole
  d Stateless Session Beans
  ```

  When you specify the maximum number of pooled Stateless Session Beans, you also need to specify the minimum number of pooled Stateless Session Beans.

9.10.3 Setting the runtime properties for Entity Beans

You set the application runtime properties for the individual Entity Beans that form an application.

(1) Attribute file to be edited

Entity Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an Entity Bean attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
  cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
  -name -c path-of-the-Entity-Bean-attribute-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name account -type ejb -resname account/MyAccoun
  t -c C:\home\MyAccount.xml
  ```

- Setting the attributes
  Execute the following command to apply the values of the Entity Bean attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
  cation-name -type ejb -resname EJB-JAR-display-name/Entity-Bean-display
  -name -c path-of-the-Entity-attribute-file
  ```

Example of execution

```
cjsetappprop MyServer -name account -type ejb -resname account/MyAccoun
t -c C:\home\MyAccount.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for the runtime attributes of Entity Bean (<runtime>):

Items	Mandatory	Corresponding tags
Look up name of the Enterprise Bean#1	Y	<lookup-name>
Optional name of the Enterprise Bean name having a remote interface#1	O	<optional-name>
Optional name of the Enterprise Bean name having a local interface#1	O	<local-optional-name>
Maximum number of sessions#2	O	<maximum-instances>
Pass by reference settings#3	O	<pass-by-reference>
Maximum number of the instances in the pool	Y	<pooled-instance> - <maximum>
Minimum number of the instances in the pool	Y	<pooled-instance> - <minimum>
Data caching method	Y	<caching-model>
Existence period of EJB object	O	<entity-timeout>

Legend:

Y: Mandatory

O: Optional

#1

For referencing and changing the name registered in the JNDI namespace, see *9.13.2 Referencing and changing the Enterprise Bean name*.

#2

The maximum number of sessions specified in the Entity Bean is the maximum number of Entity Beans that can be used from the EJB client application. If '0' is specified, the number of remote objects of the Entity Bean that can be generated concurrently becomes infinite (the real maximum value is 2147483647).

#3

The pass by reference settings can also be specified in the `usrconf.properties` file. If the settings are specified in either the properties or in the `usrconf.properties` file, pass by reference settings become valid.

For details on the property settings, see *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

The property settings and operations are explained below:

Maximum number of instances in the pool

The Entity Beans pooled in the memory have the following two states:

- ready state

  An Entity Bean in a ready state indicates a bean in a state where the data is read from the database during an instance and has an identity as an Entity Bean. The beans in the ready state are already executable when accessed from the EJB client application.

- pool state

An Entity Bean in a pool state indicates a bean in a state where the data is not read from the database during an instance and does not have an identity as an Entity Bean. Once the beans in the pool state are activated to a ready state, the beans can be executed.

If there are several Entity Beans in the ready state, some of these Entity Beans are passivated to the pool state. At this time, however, among the Entity Beans in the ready state, the beans that are in transaction are not passivated.

Specify the maximum number of pooled Entity Beans that are in the ready state and pool state, in the maximum number (<pooled-instance> - <maximum>) of the instances in the pool. This is the upper limit of the Entity Bean instances deployed on the memory.

If '0' is specified in the maximum number of instances in the pool, the number of pooled Entity Bean instances becomes infinite.

If the number of pooled instances exceeds the maximum number of instances in the pool, J2EE server passivates the instances that are least activated.

Minimum number of instances in the pool

Specify the minimum number of pooled Entity Beans that are in the ready state and pool state. This value is the lower limit of the Entity Bean instances deployed on the memory. The number of Entity Beans specified here is generated when the J2EE application starts. These Entity Beans are converted to the pool state or the ready state and are then pooled. The number of pooled Entity Beans lies between the maximum and the minimum count, depending on the volume of access from the EJB client applications.

If '0' is specified in the minimum number (<pooled-instance> - <minimum>) of instances in the pool, the Entity Bean is not generated when the J2EE application starts. Specify the minimal memory size in which the Entity Bean instances, not referenced by the client, are to be stored. This value must not exceed the maximum number of instances in the pool (<pooled-instance> - <maximum>).

Data caching method

Specify the cache model (*commit option*) of the Entity Bean.

- Full caching (commit option A)

  When the transaction starts, the data from the database is not read in the Entity Bean instance; therefore, the transaction is started with the Entity Bean in the same state as when the transaction was committed previously (For example, if another J2EE server updates the Entity Bean between the previous commit transaction and the start of the transaction, the consistency in the Entity Bean state is not maintained). Full caching is a cache model for the reference node Entity Bean.

- Caching (commit cache option B)

  When the transaction starts, the data from the database is read in the Entity Bean instance; therefore, the transaction is started with the Entity Bean in the same state as the latest state of the database. Caching is a cache model for the update node Entity Bean.

- No caching (commit cache option C)

  The Entity Bean is passivated when the transaction is committed. When the transaction starts, the Entity Bean is activated once and the data from the database is read in the Entity Bean instance. The transaction is, therefore, started with the Entity Bean in the same state as the latest state of the database. No caching is a cache model for the update node Entity Bean. Furthermore, since these Entity Beans are definitely passivated upon transaction commit, this cache model suits the case where you use several Entity Beans.

Existence period of the EJB object

Specify the timeout value in seconds. Specify the value as 0 or as a positive integer. At the minimum, the EJB object of the passivated Entity Bean exists for the time period specified in this timeout value. At the maximum, the EJB object of the passivated Entity Bean only exists for the time period specified in this timeout value + the time specified in ejbserver.container.passivate.scan.interval of the `usrconf.properties` file. If the client does not access the EJB object even after the timeout period elapses, the applicable EJB object is deleted.

If '0' is specified in the existence period of the EJB object (<entity-timeout>), timeout does not occur.

(4) Notes

- If an upper limit is specified for the maximum number of EJB objects of the Entity Bean that can be generated concurrently, an upper limit must also be specified for the maximum number of pooled Entity Beans.
- Specify the maximum number of EJB objects of the Entity Beans that can be generated concurrently, the maximum number of pooled Entity Beans and the minimum number of pooled Entity Beans such that the following conditions are fulfilled:

```
Minimum number of pooled Entity Beans ≤ Maximum number of pooled Entity Be
ans ≤ Maximum number of EJB objects of the Entity Beans that can be genera
ted concurrently
```

9.10.4 Setting the runtime properties for Message-driven Beans

You set the application runtime properties for the individual Message-driven Beans that form an application.

(1) Attribute file to be edited

Message-driven Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a Message-driven Bean attribute file:

  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type ejb -resname EJB-JAR-display-name/Message-driven-Bean
-display-name -c path-of-the-Message-driven-Bean-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name message -type ejb -resname message/MyMessag
eBean -c C:\home\MyMessageBean.xml
```

- Setting the attributes
  Execute the following command to apply the values of the Message-driven Bean attribute file:

  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE appli
cation-name -type ejb -resname EJB-JAR-display-name/Message-driven-Bean
-display-name -c path-of-the-Message-driven-Bean-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name message -type ejb -resname message/MyMessag
eBean -c C:\home\MyMessageBean.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for the runtime attributes of Message-driven Bean (<runtime>):

Item	Mandatory	Corresponding tag
Maximum number of instances in the pool#	Y	<pooled-instance> - <maximum>

Legend:

Y: Mandatory

#

During the initialization of the Server Session pool, the number of Beans up to the specified value are generated. You can specify a value from 1 to 2147483647 (maximum value of int type). The default value is 1. In order to generate the specified number of JMS sessions, define the JMS sessions as per the number of JMS sessions that can be generated in the JMS provider.

For details on the property settings, see *3.6.1 Specifications of the HITACHI MessageDrivenBean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.11 Defining the runtime attributes of servlets and JSPs

You define the runtime attributes of the Web applications (servlets and JSPs) that form a J2EE application. The definitions include the following:

- Defining the context root of a J2EE application
- Defining the control of the number of threads executed concurrently for each Web application
- Defining the control of the number of threads executed concurrently for each URL group
- Defining the monitoring of the number of requests pending for each URL group

The following subsections describe the properties to be set with the respective definitions:

9.11.1 Defining the context root of a J2EE application

You define the context root of a J2EE application.

You can also define the root context in the context root. *Root context* is a context of a context root with blank characters "". If a `welcome` file is created in the root context, the home page of the J2EE application can be displayed from a URL with only the domain name, like *http: //www.cosminexus.com/*.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a WAR attribute file:
  Execution format
  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
  bute-file
  ```
  Example of execution
  ```
  cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
  me\adder_war.xml
  ```
- Setting the attributes
  Execute the following command to apply the values of the WAR attribute file:
  Execution format
  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
  bute-file
  ```
  Example of execution
  ```
  cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
  me\adder_war.xml
  ```

(3) Attribute settings to be edited

The following table describes the property settings for context root definition (<runtime>) of the J2EE application:

Item	Mandatory	Corresponding tag
Context root	Y	<context-root>

Legend:
Y: Mandatory

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(4) Notes

- Do not specify strings beginning with "ejb/", or "web/" in the context root.
  If the components of the context root path in multiple J2EE applications have an inclusion relation (for example: "test" and "test/jsp") and if a URL (for example: "/test/jsp/test.jsp") containing the path to be included is specified and accessed, then the application (in the example, path that has "test/jsp" in the context root) of the path to be included is enabled and the application of the included path cannot be accessed.
- Context root is used as the directory name in the working directory. Specify the context root so that the path length of the working directory does not reach the upper limit set for the platform. For details on estimating the path length of the working directory, see *C.1 Work directory of the J2EE server* in the *uCosminexus Application Server System Setup and Operation Guide*.
- In the Web application that is started by using the root context, do not use a configuration in which the URL begins with "ejb" or "web".
- In the messages that are output to the console and the log file, the context root of root context is displayed by blank characters "".

9.11.2 Defining the control of the number of threads executed concurrently for each Web application

You specify the definition in order to control the number of threads executed concurrently in the Web container.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire a WAR attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

- Setting the attributes

  Execute the following command to apply the values of the WAR attribute file:

  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for controlling the number of threads executed concurrently in the Web container (<thread-control>):

Items	Mandatory	Corresponding tags
Maximum number of threads executed concurrently	Y	<thread-control-max-threads>
Number of exclusive threads	Y	<thread-control-exclusive-threads>
Size of the pending queue of each Web application	Y	<thread-control-queue-size>

Legend:

Y: Mandatory

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.11.3 Defining the control of the number of threads executed concurrently for each URL group

You specify the definition in order to control the number of threads executed concurrently for each URL group.

To enable the control of number of threads executed concurrently for each URL group, specify the definition for controlling the number of threads executed concurrently for each Web application. For more details on controlling of the number of threads executed concurrently for each Web application, see *9.11.2 Defining the control of the number of threads executed concurrently for each Web application*.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file

Execute the following command to acquire a WAR attribute file:

Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

- Setting the attributes

  Execute the following command to apply the values of the WAR attribute file:

  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for controlling the number of threads executed concurrently for each URL group (<urlgroup-thread-control>):

Items	Mandatory	Corresponding tags
Definition name	Y	<urlgroup-thread-control-name>
Maximum number of threads executed concurrently	Y	<urlgroup-thread-control-max-threads>
Number of exclusive threads	Y	<urlgroup-thread-control-exclusive-threads>
Size of the pending queue of each URL group	Y	<urlgroup-thread-control-queue-size>
URL pattern	Y	<urlgroup-thread-control-mapping> - <url-pattern>

Legend:

Y: Mandatory

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(4) Notes

- The number of threads executed by controlling the number of threads executed concurrently for each URL group, is included in the number of threads executed for each Web application. As a result, when one thread is executed for each URL group, one thread is also executed for each Web application in this Web application.
- Set the number of threads executed concurrently for each URL group in the following range:
  - Maximum number of threads executed concurrently

The range for specifying the maximum number of threads executed concurrently is as follows:

```
1 ≤ Sum total of the maximum number of threads executed concurrently fo
r each URL group ≤ Maximum number of threads executed concurrently for
each Web application
```

- Number of exclusive threads

  The range for specifying the number of exclusive threads is described below.

  Both condition 1 and condition 2 need to be fulfilled.

```
Condition 1:
When the maximum number of threads executed concurrently for each Web a
pplication = Number of exclusive threads for each Web application
  0 ≤ Number of exclusive threads for each URL group ≤ Maximum number o
f threads executed concurrently for each URL group
When maximum number of threads executed concurrently for each Web appli
cation ≠ Number of exclusive threads for each Web application
  0 ≤ Number of exclusive threads for each URL group < Maximum number o
f threads executed concurrently for each URL group

Condition 2:
  Sum total of the number of exclusive threads for each URL group ≤ Num
ber of exclusive threads for each Web application
```

9.11.4 Defining and monitoring the number of requests pending for each URL group

You specify the definition for monitoring the number of requests pending for each URL group.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file

  Execute the following command to acquire a WAR attribute file:

  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

- Setting the attributes

  Execute the following command to apply the values of the WAR attribute file:

Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

(3) Attribute settings to be edited

The following table describes the property settings for monitoring the number of requests pending for each URL group (<urlgroup-thread-control> - <stats-monitor> - <waiting-request-count>):

Items	Mandatory	Corresponding tags
Existence of threshold value event monitoring	Y	<enabled>
Output upper threshold value of threshold value event#	Y	<high-threshold>
Output lower threshold value of threshold value event#	Y	<low-threshold>

Legend:

Y: Mandatory

#

Specify such that 'Upper threshold value that outputs the threshold value event ≥ Lower threshold value that outputs the threshold value event'.

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.12 Setting the default character encoding

9.12.1 Setting method

You set the default character encoding for each Web application.

You can set the default encoding for the following character encoding:

- Character encoding of the request body and query
- Character encoding of the response body
- Character encoding of JSP files

(1) Property files to be edited

WAR property files

(2) Acquiring the property files to be edited and setting the properties

- Acquiring the property files
  Execute the following command to acquire WAR property files:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjgetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

- Setting the property files
  Execute the following command to apply the values of WAR property files:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjsetappprop MyServer -name adder -type war -resname adder_war -c C:\ho
me\adder_war.xml
```

(3) Settings of the property to be edited

The following table describes the property settings of the default character encoding:

Item	Mandatory	Corresponding tag name
Character encoding of the request body and query	O	<http-request> - <encoding>

Item	Mandatory	Corresponding tag name
Character encoding of the response body	O	<http-response> - <encoding>
Character encoding of JSP files	O	<jsp> - <page-encoding>

Legend:

O: Optional

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.13 Referencing and changing the names registered in JNDI namespace

You change the names of the J2EE applications and the Enterprise Beans registered in the JNDI namespace. You can also add an optional name for the Enterprise Bean name.

An *EJB Home object* is assigned to the JNDI namespace (HITACHI_EJB/SERVERS/*server-name*/*EJB*/*J2EE application-name*/*Enterprise-Bean-name*) when the J2EE application starts. Use this name when referencing from the EJB client application and when referencing by using the name switching functionality.

An *Optional name* is added in the EJB home object. By adding the optional name, the EJB client application is able to get the EJB home object with any name from the JNDI namespace. This functionality is called the *user-specified namespace functionality*.

For the JNDI name space of EJB home objects, see *2.3 Binding and lookup of objects to the JNDI name space* in the *uCosminexus Application Server Common Container Functionality Guide*. For the user-specified name space functionality, see *2.6 Adding optional names to Enterprise Beans or J2EE resources (user-specified name space functionality)* in the *uCosminexus Application Server Common Container Functionality Guide*.

9.13.1 Referencing the J2EE application names

The procedure for referencing the J2EE application name is as follows:

1. Acquire the application attribute file.
   Execute the following command to acquire an application attribute file:
   Execution format
   ```
   cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applicati
   on-name -c path-of-the-application-attribute-file
   ```
   Example of execution
   ```
   cjgetappprop MyServer -name account -c C: \home\accountAPP.xml
   ```
2. Open the application attribute file with the text editor.
   Reference the J2EE application name (<lookup-name>) of the application attribute file.

For the `cjgetappprop` command, see *cjgetappprop (get HITACHI Application Property)* in the *uCosminexus Application Server Command Reference Guide*. For details on the property files, see *3.2 HITACHI Application Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

Note:
The J2EE application name registered in JNDI namespace is assigned automatically based on the J2EE application name and this name cannot be changed.

9.13.2 Referencing and changing the Enterprise Bean name

You reference and change the Enterprise Bean name of the following Enterprise Beans:

- Session Bean
- Entity Bean

(1) Attribute files to be edited

- Session Bean attribute file
- Entity Bean attribute file

(2) Acquiring the attribute files to be edited and setting the attributes

- Acquiring the attribute files

  Execute the following command to acquire an Enterprise Bean attribute file (Session Bean attribute file or Entity Bean attribute file):

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute-file#
  ```

  #

  Specify the file path of the Session Bean attribute file or the Entity Bean attribute file in the path of the Enterprise Bean attribute file.

  Example of execution

  ```
  cjgetappprop MyServer -name account -type ejb -resname account/MyAccoun
  t -c C:\home\MyAccount.xml
  ```

- Setting the attribute files

  Execute the following command to apply the values of the Enterprise Bean attribute file (Session Bean attribute file or Entity Bean attribute file):

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
  lay-name -c path-of-the-Enterprise-Bean-attribute-file#
  ```

  #

  Specify the path of the Session Bean attribute file or the Entity Bean attribute file in the path of the Enterprise Bean attribute file.

  Example of execution

  ```
  cjsetappprop MyServer -name account -type ejb -resname account/MyAccoun
  t -c C:\home\MyAccount.xml
  ```

(3) Attribute settings to be edited

Reference and change the Enterprise Bean name of the Enterprise Bean attribute file (Session Bean attribute file or Entity Bean attribute file) with the runtime attributes (<runtime>).

Items	Mandatory	Corresponding tags
Lookup name of the Enterprise Bean	Y	<lookup-name>
Optional name of the Enterprise Bean name having a remote interface	O	<optional-name>
Optional name of the Enterprise Bean name having a local interface	O	<local-optional-name>

Legend:

Y: Mandatory

O: Optional

For property settings, reference the following subsections:

- *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*

(4) Notes

- You can specify the value only when `true` is set in the ejbserver.cui.optionalname.enabled key of `usrconf.properties` file and EJB implements a remote interface.
- The initial value of the Enterprise Bean name registered in the JNDI namespace uses the ejb-name of an Enterprise Bean.
- The Enterprise Bean name registered in the JNDI namespace does not exist in the Message-driven Bean.

9.14 CTM scheduling

You specify the settings for scheduling when you are using CTM. You can use CTM only in the products having Cosminexus Component Transaction Monitor in the component software. For details on the products that you can use, see *2.2.1 Correspondence of the products and component software* in the manual *uCosminexus Application Server Overview*.

Specify CTM scheduling as follows:

- Specify scheduling for applications.
- Specify scheduling for Stateless Session Beans that you want to schedule.

9.14.1 Scheduling the J2EE applications

Specify the settings for integrating each J2EE application with CTM.

You must set up both the Application properties and Session Bean Properties. Hitachi recommends that you use the HITACHI Application Integrated Property File to specify the settings.

(1) Attribute files to be edited

HITACHI Application Integrated Property File

(2) Acquiring the attribute files to be edited and setting the attributes

- Acquiring the attribute files
  You execute the following command to acquire the HITACHI Application Integrated Property File:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
ation-name -type all -c path-of-the-hitachi-application-integrated-prop
erty-file
```

  Example of execution

```
cjgetappprop MyServer -name account -type all -c C:\home\account.xml
```

- Setting the attributes
  You execute the following command to apply the values of the HITACHI Application Integrated Property File:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
ation-name -type all -c path-of-the-hitachi-application-integrated-prop
erty-file
```

  Example of execution

```
cjsetappprop MyServer -name account -type all -c C:\home\account.xml
```

(3) Attribute settings to be edited

The attribute settings to be edited are as follows:

- Application properties
- Session Bean attribute file

The individual settings are as follows:

(a) Application properties

The following table describes the settings for integrating each J2EE application with CTM:

Items	Mandatory	Corresponding tags
Presence of CTM integration	Y	<managed-by-ctm>
Queue deployment model	Y	<scheduling-unit>
Queue name	Y	<scheduling> - <queue-name>
Number of threads	Y	<scheduling> - <parallel-count>
Queue length	O	<scheduling> - <queue-length>

Legend:
  Y: Mandatory
  O: Optional

For details on the property settings, see *3.2.1 Specifications of the HITACHI Application Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Session Bean attribute file

The following table describes the runtime attribute settings (<session-runtime>) for integrating each J2EE application with CTM.

Items	Mandatory	Corresponding tags
Settings for scheduling	Y	<enable-scheduling>

Legend:
  Y: Mandatory

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(4) Notes

If you specify the settings using the application property file and Session Bean property file without using the HITACHI Application Integrated Property File, you must specify the settings in the following order:

1. Execute the following command to apply the values of the `<enable-scheduling>` tag in the Session Bean property file:

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name -type ejb -resname display-name-of-EJB-JAR/ display-name-of-Statel
ess-Session-Bean -c path-of-the-hitachi-The Session Bean property file
```

2. Execute the following command to apply the values of the tags that exist under the `<managed-by-ctm>` tag, `<scheduling-unit>` tag, and `<scheduling>` tag of the application property file:

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name -c path-of-the-hitachi-application-property-file
```

9.14.2 Scheduling the Stateless Session Beans

For using CTM to schedule the Stateless Session Beans included in a J2EE application that is to be scheduled using CTM, specify the settings for scheduling Stateless Session Beans.

You can schedule only Stateless Session Beans for which the remote home interface is implemented.

To use the scheduling function, set up the runtime properties of Stateless Session Beans. For details on setting up the runtime properties of Stateless Session Beans, see *9.10.2 Setting the runtime properties for Stateless Session Beans*.

You also specify the settings for integrating with CTM in the application properties.

You must set up both the application properties and Session Bean Properties. Hitachi recommends that you use the HITACHI Application Integrated Property File to specify the settings.

(1) Attribute files to be edited

HITACHI Application Integrated Property File

(2) Acquiring the attribute files to be edited and setting the attributes

- Acquiring the attribute files
  You execute the following command to acquire the HITACHI Application Integrated Property File:
  Execution format
  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type all -c path-of-the-hitachi-application-integrated-prop
  erty-file
  ```
  Example of execution
  ```
  cjgetappprop MyServer -name adder -type all -c C:\home\Adder.xml
  ```
- Setting the attributes
  You execute the following command to apply the values of the HITACHI Application Integrated Property File:
  Execution format
  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type all -c path-of-the-hitachi-application-integrated-prop
  erty-file
  ```
  Example of execution
  ```
  cjsetappprop MyServer -name adder -type all -c C:\home\Adder.xml
  ```

(3) Attribute settings to be edited

The attribute settings to be edited are as follows:

- Application properties
- Session Bean attributes

The individual settings are as follows:

(a) Application properties

The following table describes the settings for integrating Stateless Session Beans with CTM:

Items	Mandatory	Corresponding tags
Presence of CTM integration	Y	<managed-by-ctm>
Queue deployment model	Y	<scheduling-unit>

Legend:
Y: Mandatory

For details on the property settings, see *3.2.1 Specifications of the HITACHI Application Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(b) Session Bean attribute file

The following table describes the runtime attribute settings (<session-runtime>) for integrating Stateless Session Beans with CTM:

Items	Mandatory	Corresponding tags
Settings for scheduling	Y	<enable-scheduling>
Queue name	Y	<scheduling> - <queue-name>
Number of threads	Y	<scheduling> - <parallel-count>
Queue length	O	<scheduling> - <queue-length>

Legend:
Y: Mandatory
O: Optional

For details on the property settings, see *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

(4) Notes

If you specify the settings using an application property file and Session Bean property file and without using the HITACHI Application Integrated Property File, you must specify the settings in the following order:

1. Execute the following command to apply the values of the `<enable-scheduling>` tag in the Session Bean property file:

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name -type ejb -resname display-name-of-EJB-JAR/ display-name-of-Statel
ess-Session-Bean -c path-of-the- Session Bean property file
```

2. Execute the following command to apply the values of the `<managed-by-ctm>` tag and `<scheduling-unit>` tag in the application property file:

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name -c path-of-the-HITACHI-application-property-file
```

3. Execute the following command to apply the values of the tags that exist under the `<scheduling>` tag of the Session Bean property file :

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applicati
on-name -type ejb -resname display-name-of-EJB-JAR/ display-name-of-Statel
es-Session-Bean -c path-of-the-hitachi-The Session Bean property file
```

9.15 Setting invocation order

You set up the invocation order for the J2EE applications and the Enterprise Bean and WAR included in the J2EE applications.

9.15.1 Setting the invocation order for J2EE applications

Set the invocation order for the J2EE applications.

(1) Attribute file to be edited

Application attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an application attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -c path-of-the-application-attribute-file
```

  Example of execution

```
cjgetappprop MyServer -name account -c C:\home\MyAccount.xml
```

- Setting the attributes
  Execute the following command to apply the values of the application attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -c path-of-the-application-attribute-file
```

  Example of execution

```
cjsetappprop MyServer -name account -c C:\home\MyAccount.xml
```

(3) Attribute settings to be edited

The following table describes the invocation order settings for J2EE applications:

Item	Mandatory	Corresponding tag
Start and stop order#	O	<start-order>

Legend:
O: Optional

#

The smaller the value, the earlier the application will start and larger the value, the earlier the application will stop. Note that if you specify the same value for multiple J2EE applications, the invocation order between the applications may not be as desired.

For details on the property settings, see *3.2.1 Specifications of the HITACHI Application Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.15.2 Setting the invocation order for Enterprise Beans

Set up the invocation order for the Enterprise Beans.

(1) Attribute file to be edited

- Session Bean attribute file
- Entity Bean attribute file
- Message-driven Bean attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire an Enterprise Bean attribute file that is to be edited:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-disp
lay-name -c path-of-the-Enterprise-Bean-attribute-file#
```

  #
  Specify the path of an attribute file that is to be edited in the path of the Enterprise Bean attribute file.
  Example of execution

```
cjgetappprop MyServer -name account_eb -type ejb -resname account/accou
nt -c C:\home\MyAccount.xml
```

- Setting the attributes
  Execute the following command to apply the values of the edited Enterprise Bean attribute file:
  Execution format

```
jsetappprop [server-name] [-nameserver provider-URL] -name J2EE applica
tion-name -type ejb -resname EJB-JAR-display-name/Enterprise-Bean-displ
ay-name -c path-of-the-Enterprise-Bean-attribute-file#
```

  #
  In the path of the Enterprise Bean attribute file, specify the path of the attribute file in which the values are to be applied.
  Example of execution

```
cjsetappprop MyServer -name account -type ejb -resname account/account_
eb -c C:\home\MyAccount.xml
```

(3) Attribute settings to be edited

The following table describes the invocation order settings for the Enterprise Bean:

Item	Mandatory	Corresponding tag
Start and stop order#	O	<start-order>

Legend:
O: Optional

#
The smaller the value, the earlier the application will start and larger the value, the earlier the application will stop. Note that if you specify the same value for the Enterprise Beans, the invocation order between the beans may not be as desired.

For property settings, reference the following subsections:

- *3.4.1 Specifications of the HITACHI Session Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.5.1 Specifications of the HITACHI Entity Bean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*
- *3.6.1 Specifications of the HITACHI MessageDrivenBean Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*

9.15.3 Setting the invocation order for servlets and JSPs

You set the invocation order for a WAR that contains servlets or JSPs.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire the following WAR attribute file:
  Execution format

```
cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

  Example of execution

```
cjgetappprop MyServer -name account -type war -resname account_war -c C
:\home\account_war.xml
```

- Setting the attributes
  Execute the following command to apply the values of the WAR attribute file:
  Execution format

```
cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
bute-file
```

Example of execution

```
cjsetappprop MyServer -name account -type war -resname account_war -c C
:\home\account_war.xml
```

(3) Attribute settings to be edited

The following table describes the invocation order settings for a WAR that contains servlets and JSPs:

Item	Mandatory	Corresponding tag
Start and stop order#	O	<start-order>

Legend:

O: Optional

#

The smaller the value, the earlier the application will start and larger the value, the earlier the application will stop. Note that if you specify the same value for servlets and JSPs, the invocation order between servlets and JSPs may not be as desired.

For details on the property settings, see *3.7.1 Specifications of the HITACHI WAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

9.16 Error notification settings for servlets and JSPs

9.16.1 Setting method

If an error occurs at the following points of time, specify whether to report the error and cancel the starting process of J2EE applications:

- When an error occurs in the initialization process of servlets or JSPs to be loaded (<load-on-startup> is specified), when starting the J2EE application.
- When error occurs during tag library analysis.

(1) Attribute file to be edited

WAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file
  Execute the following command to acquire the following WAR attribute file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
  bute-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name account -type war -resname account_war -c C
  :\home\account_war.xml
  ```

- Setting the attributes
  Execute the following command to apply the values of the WAR attribute file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE applic
  ation-name -type war -resname WAR-display-name -c path-of-the-WAR-attri
  bute-file
  ```

  Example of execution

  ```
  cjsetappprop MyServer -name account -type war -resname account_war -c C
  :\home\account_war.xml
  ```

(3) Attribute settings to be edited

The following table describes the error notification settings for a WAR that contains servlets or JSPs:

Item	Mandatory	Corresponding tag
Scope of error notification when starting	O	<start-notify-error>

Legend:

O: Optional

(4) Notes

- *Scope of error notification settings when the initialization of the Web application fails*

  In the J2EE server mode, you can set whether to continue the starting process of the J2EE application or whether to report the error and cancel the starting process of the J2EE application when initialization of the Web application fails. However, if an error occurs that requires the Web application to be disabled, you cannot specify the error notification settings. Regardless of the error notification settings, the starting process of the J2EE application is cancelled.

  The failure in the initialization of the Web application and the scope of error notification settings will be explained here. The following table describes the errors that occur during the initialization of the Web application and the feasibility of specifying the error notification settings of whether to continue the starting process of the J2EE application or to cancel the process as an error:

Table 9–5: Description of failure in the initialization of the Web application and the scope of error notification settings

Items	Description of initialization failure in the Web application	Scope of error notification settings
Temporary directory for JSP	When the generation of the directory fails because there is no permission to access the temporary directory for JSP.	No
	When the deletion of the directory fails because there is no permission to access the temporary directory for JSP.	Yes
`web.xml`	When an error occurs in the analysis of `web.xml`.	No
Filter	When a filter class specified in the filter-class element of `web.xml` or a class on which the filter class depends cannot be found.	No
	When the following invalid filter classes are specified in the filter-class element of `web.xml`: 1. Class that does not implement javax.servlet.Filter interface 2. Class that does not have a constructor that does not take an argument	No
	When an exception occurs during the initialization of the filter class.	No
Listener	When the listener class specified in the listener-class element of `web.xml` or a class on which the listener class depends cannot be found.	No
	When the following invalid listener classes are specified in the listener-class element of `web.xml`: 1. Class that does not implement any of the following interfaces: javax.servlet.ServletContextListener javax.servlet.ServletContextAttributeListener javax.servlet.ServletRequestListener javax.servlet.ServletRequestAttributeListener javax.servlet.http.HttpSessionListener javax.servlet.http.HttpSessionAttributeListener 2. Class that does not have a constructor that does not take an argument	No
	When an exception occurs during the initialization of the listener-class.	No
Servlet that specifies <load-on-startup> in `web.xml`	When the servlet class specified in the servlet-class element of `web.xml` or a class on which the servlet class depends cannot be found.	Yes
	When an exception occurs during the initialization of the servlet.	Yes

Items	Description of initialization failure in the Web application	Scope of error notification settings
JSP that specifies <load-on-startup> in `web.xml`	When the JSP file specified in the jsp-file element of `web.xml` cannot be found.	Yes
	When a Java source file cannot be generated from JSP because there is no permission to create the subdirectory in the temporary directory for JSP	Yes
	When a Java source file cannot be generated from JSP because there is no permission to access the subdirectory of the temporary directory for JSP	Yes
	When an error occurs when generating a Java source file from JSP.	Yes
	When an error occurs in the compilation of the source code of the servlet generated from JSP.	Yes
	In the Web applications prior to Servlet2.3, when taglib is not mapped to the <taglib> tag of `web.xml` and absolute URI is specified in the uri attribute of the taglib directive of JSP.	Yes
	When an error occurs during the analysis of the JSP document.	Yes
	When an exception occurs during the initialization of JSP.	Yes
Tag library (When executed using the extension of JSP that specifies <load-on-startup>)	When the TLD file cannot be read.	Yes
	When an error occurs during the analysis of the TLD file.	Yes
	When the tag library validator class or a class on which the tag library validator class depends cannot be found.	Yes
	When the following invalid tag library validator data classes are specified: 1. Class that does not inherit javax.servlet.jsp.tagext.TagLibraryValidator class 2. Class that does not have a constructor that does not take an argument	Yes
	When an exception occurs during the initialization of the tag library validator class.	Yes
	When an exception occurs during the verification of JSP due to the tag library validator.	Yes
	When TagExtraInfo class or a class on which the TagExtraInfo class depends cannot be found.	Yes
	When an invalid TagExtraInfo class is specified. An invalid TagExtraInfo class indicates a class specified in the Tei-class element or teiclass element and matches with one of the following conditions: • TagExtraInfo class is not inherited • Is an interface and abstract class • Does not have a public constructor	Yes
	When an exception occurs during the initialization of a class specified in tei-class element or teiclass element of TLD file.	Yes
	When the tag library validator class reports an error found during the verification of a JSP page.	Yes
	When the TagExtraInfo class reports an error found during the verification of attributes.	Yes
	When an exception occurs during the verification of attributes by the TagExtraInfo class.	Yes
	When the class defined in the function-class element of the TLD file cannot be found.	Yes

Items	Description of initialization failure in the Web application	Scope of error notification settings
	When an exception occurs while the class defined in the function-class element of the TLD file is being loaded.	Yes
	When a parameter class defined in the function-signature element of the TLD file cannot be found.	Yes
	When a method defined in the function-signature element of the TLD file cannot be found.	Yes
	When an exception occurs at the time of loading the parameter class defined in a function-signature element of a TLD file or when accessing the class specified in a function-class element.	Yes
	When the class defined in the type element of the TLD file cannot be found.	Yes
	When an exception occurs while the class defined in the type element of the TLD file is being loaded.	Yes

9.17 Interceptor settings

9.17.1 Setting method

This section describes how to set up an interceptor. If you want to use the interceptor on Cosminexus, specify the default interceptor as an EJB-JAR property.

(1) Attribute file to be edited

EJB-JAR attribute file

(2) Acquiring the attribute file to be edited and setting the attributes

- Acquiring the attribute file

  You execute the following command to acquire the EJB-JAR property file:

  Execution format

  ```
  cjgetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type ejb -resname display-name-of-EJB-JAR -c path-of-the-hi
  tachi-EJB-JAR-property-file
  ```

  Example of execution

  ```
  cjgetappprop MyServer -name adder -type ejb -resname adder -c C:\home\a
  dder_ejb.xml
  ```

- Setting the attributes

  You execute the following command to apply the values of the EJB-JAR property file:

  Execution format

  ```
  cjsetappprop [server-name] [-nameserver provider-URL] -name J2EE-applic
  ation-name -type ejb -resname display-name-of-EJB-JAR -c path-of-the-hi
  tachi-EJB-JAR-property-file
  ```

  Example of execution

  ```
  cjsetappprop MyServer -name adder -type ejb -resname adder -c C:\home\a
  dder_ejb.xml
  ```

Note:

You can acquire the EJB-JAR property file included in a running application, but you cannot apply the defined and related information of the EJB-JAR property file.

(3) Attribute settings to be edited

The following table describes the settings (`<interceptor-binding>`) for the default interceptor:

Items	Mandatory	Corresponding tags
Explanation	O	<description>
EJB name of the default interceptor (specify wildcard)	Y	<ejb-name>

Items	Mandatory	Corresponding tags
Class name of the interceptor class	O	<interceptor-class>
Setting an execution order of interceptors (Class name of the interceptor class)	O	<interceptor-order>-<interceptor-class>
Setting a call inhibit of the default interceptor	O	<exclude-default-interceptors>
Setting a call inhibit of the class level interceptor	O	<exclude-class-interceptors>
Business method name of EJB	O	<named-method>-<method-name>
Method parameters	O	<named-method>-<method-params>-<method-param>

Legend:

Y: Mandatory

O: Optional

For details on the property settings, see *3.3.1 Specifications of the HITACHI EJB-JAR Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For the method to set interceptors and notes on interceptors, see *2.15 Using interceptors* in the *uCosminexus Application Server EJB Container Functionality Guide*.

9.18 Setting other properties

For details on the other properties to be set when creating and customizing J2EE applications, see the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

10 Executing J2EE Applications

This chapter describes the execution of J2EE applications with the server management commands.

10.1 Overview of J2EE application execution

After finishing the definition of properties and operations required for J2EE applications, you can execute the J2EE applications.

The following table provides an overview of the execution of J2EE applications:

Table 10–1: Overview of the execution of J2EE applications

Settings	Contents	J2EE application format		Reference
		Archive format	Exploded archive format	
Starting and stopping J2EE applications	Start J2EE applications normally or by compiling JSP. Stop J2EE applications normally or by forced termination.	Y	Y	10.2
Referencing the list of J2EE applications	Reference the list of imported applications.	Y	Y	10.3
Deleting J2EE applications	Delete J2EE applications.	Y	R#1	10.4
Switching between J2EE applications	Switch the following J2EE applications on the J2EE server: • Applications in the archive format • Applications in the exploded archive format	R#2	R#2	10.5
Changing the J2EE application name	Change the name of a J2EE application.	Y	Y	10.6
Acquiring the RMI-IIOP stubs and interfaces	Acquire RMI-IIOP stubs and interfaces of J2EE applications.	Y	Y	10.7
Referencing the list of transactions	Reference a list of transaction information. When a J2EE server is running, transaction information like the operation state and the time elapsed since starting the server is displayed. When a J2EE server is being stopped, the state of an unprocessed transaction pending in the status file is displayed.	Y	Y	10.8

Legend:

Y: Can be executed

R: Restricted execution

#1

You cannot delete a component from the application.

Note that filter addition can be executed because filter addition is a change in the application attributes.

#2

Use the following commands to switch between J2EE applications:

- The `cjreplaceapp` command for the archive format
- The `cjreloadapp` command for the exploded archive format

10.2 Starting and stopping J2EE applications

This section describes the process of starting and stopping J2EE applications.

10.2.1 Starting J2EE applications

This subsection describes the following methods for starting J2EE applications:

- Starting J2EE applications normally
- Starting J2EE applications by compiling JSP

(1) Starting J2EE applications normally

Execute the following command to start a J2EE application. If resource adapters are included in the J2EE application, all the resource adapters included in the J2EE application will also start automatically.

Execution format

```
cjstartapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name
```

Example of execution

```
cjstartapp MyServer -name account
```

For details on the `cjstartapp` command, see *cjstartapp (start J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- When a J2EE application references J2EE resource adapters, you first start the J2EE resource adapters, and then start the J2EE application.
- You cannot control the starting order of resource adapters included in a J2EE application.
- When invoking Enterprise Beans included in other J2EE applications, start the called J2EE application and then start the calling J2EE application.

(2) Starting J2EE applications by compiling JSP

To start a J2EE application, compile all the JSP files included in the J2EE application.

Execute the following command:

Execution format

```
cjstartapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -jspc
```

Example of execution

```
cjstartapp MyServer -name App1 -jspc
```

For details on the `cjstartapp` command, see *cjstartapp (start J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

In the case of applications in the archive format, the JSP compilation results that are generated by JSP pre-compile when starting the application, are deleted when the application terminates.

For using the JSP compilation results that are generated by JSP pre-compile when starting the application, even after the application terminates, execute the following procedures:

1. Specify JSP pre-compile and start the application.
2. Export the application.
3. Use functionality like the redeploy functionality to switch to an application that contains JSP compilation results.
   For switching between the J2EE applications, see *10.5 Switching between J2EE applications*.

Furthermore, after executing JSP pre-compiling at the start of the application, if the application fails to start, the application terminates and the JSP compilation results are deleted. Before executing JSP pre-compiling at the start of the application, confirm that the application can start normally.

10.2.2 Stopping J2EE applications

This subsection describes the normal termination and the forced termination of J2EE applications.

(1) Normal termination of J2EE applications

Execute the following command to terminate a J2EE application normally. If resource adapters are included in the J2EE application, all the resource adapters included in the J2EE application will also stop automatically.

Execution format

```
cjstopapp [server-name] [-nameserver provider URL] -name J2EE-application
-name
```

Example of execution

```
cjstopapp MyServer -name account
```

For details on the `cjstopapp` command, see *cjstopapp (stop J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- When a J2EE application references resource adapters deployed as J2EE resource adapters, you first stop the J2EE application, and then stop J2EE resource adapters.
- You cannot control the stopping order of resource adapters included in a J2EE application.
- When invoking Enterprise Beans included in other J2EE applications, stop the calling J2EE application and then stop the called J2EE application.
- In the case of applications in the archive format, the compilation results generated by JSP pre-compile when starting the application, are deleted when the application terminates.

(2) Forced termination of J2EE applications

This subsection explains the forced termination of the J2EE applications.

The two methods to terminate J2EE applications forcibly are as follows:

- Execute forced termination when a J2EE application does not stop by normal termination.
- Specify the option for automatic execution of forced termination when a J2EE application does not stop and terminate the application normally.

The procedures for each forced termination are as follows:

(a) Executing forced termination when a J2EE application does not stop by normal termination

When a J2EE application does not stop even when terminated normally as per the procedures in *(1) Normal termination of J2EE applications*, you can terminate the J2EE application forcibly using the following procedures:

Note that you execute forced termination of J2EE applications only when J2EE applications do not stop properly even by normal termination.

Execute the following command to terminate a J2EE application forcibly:

Execution format

```
cjstopapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -cancel
```

Example of execution

```
cjstopapp MyServer -name account -cancel
```

For details on the `cjstopapp` command, see *cjstopapp (stop J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

(b) Specifying the automatic forced termination option of the J2EE application and executing normal termination

When executing normal termination of a J2EE application, you can specify that forced termination will be executed automatically after a timeout ends if the J2EE application does not stop properly.

Execute the following command so that forced termination will be automatically performed for a J2EE application that does not stop normally:

Execution format

```
cjstopapp [server-name] [-nameserver provider-URL] -name J2EE-application
-name -t timeout-period (seconds) -force
```

Example of execution

```
cjstopapp MyServer -name account -t 120 -force
```

For details on the `cjstopapp` command, see *cjstopapp (stop J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

10.3 Referencing the list of J2EE applications

Execute the following command to reference the list of imported J2EE applications:

If the J2EE application is in the exploded archive format, the application directory path is displayed.

Execution format

```
cjlistapp [server-name]
```

Example of execution

```
cjlistapp MyServer
```

For details on the `cjlistapp` command, see *cjlistapp (list applications)* in the *uCosminexus Application Server Command Reference Guide*.

10.4 Deleting J2EE applications

To delete J2EE applications, execute the following command:

Execution format

```
cjdeleteapp [server-name] [-nameserver provider-URL] -name J2EE-applicatio
n-name
```

Example of execution

```
cjdeleteapp MyServer -name account
```

For details on the `cjdeleteapp` command, see *cjdeleteapp (delete J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

When deleting a J2EE application that is in the exploded directory format, the application directory of the J2EE application that you want to delete is not deleted. The registration in the J2EE server is cancelled.

10.5 Switching between J2EE applications

The methods for switching the J2EE applications running in the J2EE server mode are as follows:

- Re-deploy the J2EE application.
  Delete the J2EE applications from the J2EE server, switch with another J2EE application and then start the application.
- Use the redeploy functionality.
  You can use this functionality when the J2EE application is in the archive format.
- Use the reload functionality.
  You can use this functionality when the J2EE application is in the exploded archive format.

This section describes the following methods of switching between the J2EE applications:

- Switching between archive format J2EE applications with the redeploy functionality
- Switching between exploded archive format J2EE applications with the reload functionality

10.5.1 Applications in archive format

The procedure for using the redeploy functionality to switch the archive format J2EE applications, imported into the J2EE server, to another J2EE application are as follows:

1. Compile the Java program to be updated.
2. Regenerate the EJB-JAR and WAR files.
3. Regenerate the EAR files.
4. Execute the `redeploy` command.

Execute the following *redeploy command* to switch the archive format J2EE applications (EAR files):

Execution format

```
cjreplaceapp [server-name] [-nameserver provider-URL] -name application-na
me -f EAR- file- path [-t timeout-period (seconds)] [-replaceDD]
```

Example of execution

```
cjreplaceapp MyServer -name App1 -f App1.ear -t 120
```

When switching between the J2EE applications, all the attributes of the J2EE applications can be inherited or only the runtime attributes (standalone definitions of attribute files) can be inherited. To inherit the runtime attributes only, specify the `-replaceDD` option.

For details on the `cjreplaceapp` command, see *cjreplaceapp (replace application)* in the *uCosminexus Application Server Command Reference Guide*.

Note:

- You can perform this operation regardless of whether the J2EE application is running or has stopped. When switching a running J2EE application, if you execute the `cjreplaceapp` command, the J2EE application that is being processed is stopped once and is restarted after switching.

  Note that when a running J2EE application is not terminated normally, the J2EE application is terminated forcibly. If you specify the timeout period in the `-t` option, you can set the time when the J2EE application will move to forced termination. When the timeout period is not specified in the `-t` option, the application will move to forced termination after 60 seconds.
- In the switching of applications where JSP pre-compile is executed, if you want to use the JSP pre-compile functionality even after switching the applications, the JSP compilation result must be included in the applications to be switched. If the JSP compilation result is not included in the applications to be switched, run JSP pre-compile for each Web application and include the JSP compilation result in the applications to be switched.
- When you use the redeploy functionality in the applications that contain `cosminexus.xml`, the definition information of `cosminexus.xml` before switching is destroyed and restored to the default value, and the definition information of `cosminexus.xml` after switching is overwritten. Therefore, the definition information of each Cosminexus Application Server instance that is changed using server management commands is lost in the execution environment.

10.5.2 Applications in exploded archive format

If you use the reload functionality, the regeneration of applications (creation of EAR files) is no longer required and you only need to replace the class files directly. Therefore, the process of switching between applications becomes easier.

You can execute the reload functionality only for those J2EE applications that are in exploded archive format and are in running state or in stopped state due to reload failure.

The two ways of implementing the reload functionality are as follows:

- Reload functionality based on auto-detection of updates by the J2EE server
- Reload functionality based on command execution

This section describes the reload functionality based on command execution.

The procedures for switching between J2EE applications using the reload functionality are described below:

1. Compile the Java program to be updated.
2. Execute the `reload` command.

Execute the following *reload command* to switch the J2EE applications that are in exploded archive format:

(1) Execution format

```
cjreloadapp [server-name] -name Application-name [-t timeout-period-until-fo
rced-reloading-starts]
```

(2) Example of execution

```
cjreloadapp MyServer -name App1
```

For details on the `cjreloadapp` command, see *cjreloadapp (reload application)* in the *uCosminexus Application Server Command Reference Guide*.

For details on the reload functionality and the reload functionality based on update detection, see *15.8 Detecting updates and reload of J2EE applications* in the *uCosminexus Application Server Common Container Functionality Guide*.

(3) Note

- By default, the reload functionality is disabled. To use the reload functionality, specify the property setup file (`usrconf.properties`) key as follows:
  - Setting the scope for reload functionality
    ejbserver.deploy.context.reload_scope=app
  - Setting the update detection interval
    ejbserver.deploy.context.check_interval=1

  For details on reload settings and update detection of J2EE applications, see *15.8.12 Settings for detecting updates and reload of J2EE applications* in the *uCosminexus Application Server Common Container Functionality Guide*.
- For J2EE applications including resource adapters, the RAR files are saved in the archive format. Even if one RAR file existing under an application directory is changed, the reload functionality will not be executed.
- When the reload functionality is used for applications that contain `cosminexus.xml`, the definition information of `cosminexus.xml` will not be reloaded.

10.6 Changing the J2EE application name

You change the names of the J2EE applications that are imported into the J2EE server.

To change the J2EE application names, you need to stop the J2EE applications. For more details on how to stop the J2EE applications, see *10.2.2 Stopping J2EE applications*.

Execute the following command to change the J2EE application names:

Execution format

```
cjrenameapp [server-name] [-nameserver provider-URL] -name application-nam
e -newname changed-application-name
```

Example of execution

```
cjrenameapp MyServer -name App1 -newname App2
```

For details on the `cjrenameapp` command, see *cjrenameapp (rename application)* in the *uCosminexus Application Server Command Reference Guide*.

10.7 Acquiring the RMI-IIOP stubs and interfaces

10.7.1 Command syntax and description

You acquire the RMI-IIOP stubs and interfaces of J2EE applications that are imported into the J2EE server.

The RMI-IIOP stubs and interfaces of J2EE applications that are imported into the J2EE server cannot be acquired if the J2EE application has not been started even once.

Moreover, if Enterprise Beans containing the remote interface do not exist in the J2EE application, acquiring the RMI-IIOP stubs and interfaces results in an error.

Execute the following command to acquire the RMI-IIOP stubs and interfaces:

(1) Execution format

```
cjgetstubsjar [server-name] [-nameserver provider-URL] -name application-nam
e -d storage-path-of-RMI-IIOP-stubs-and-interfaces
```

(2) Example of execution

```
cjgetstubsjar MyServer -name App1 -d temp
```

(3) Note

- You can use this functionality only if the following conditions are fulfilled:
  - If the version of the application server has been upgraded and J2EE applications that were created before the version was upgraded exist on the J2EE server
  - If the name is not changed with the `cjrenameapp` command
- You cannot acquire the RMI-IIOP stubs and interfaces based on `index.html` in the following cases:
  - If either the J2EE applications with runtime information or J2EE applications without runtime information are imported
  - If new J2EE applications are created
  - If the name is changed with the `cjrenameapp` command
- When you change the name of the application for which `application.xml` is omitted, an `application.xml` is created. Therefore, the J2EE application for which the application name is changed will be same as the application with `application.xml`.

For details on the `cjgetstubsjar` command, see *cjgetstubsjar (get RMI-IIOP stub and interface for application)* in the *uCosminexus Application Server Command Reference Guide*.

10.8 Referencing the list of transactions

10.8.1 Command syntax and description

Here, you reference a list of transaction information. You can reference the transaction information on the running J2EE server and the information about unprocessed transactions that remain in a stopped J2EE server.

(1) Displaying transaction information of a running J2EE server

You reference a list of transaction information, such as the operation state and time elapsed since starting the server.

(a) Execution format

```
cjlisttrn [server-name] [-nameserver provider-URL] [-gid global-transaction
- ID] [-pending] [-time expired-time] [-bqual]
```

(b) Example of execution

An example display of the branch ID of the entire Xid related to the transaction:

```
cjlisttrn MyServer -bqual
```

For details on the `cjlisttrn` command, see *cjlisttrn (display information about transactions operating in J2EE server)* in the *uCosminexus Application Server Command Reference Guide*.

(2) Displaying transaction information of a stopped J2EE server

You reference a list of information about unprocessed transactions that remain in the status file of a stopped J2EE server.

Execute the following command to reference the list of transactions:

(a) Execution format

```
cjlisttrnfile [server-name] [-nameserver provider-URL] [-gid global- transac
tion-ID] [-bqual]
```

(b) Example of execution

Example showing the information related to the specified global transaction ID:

```
cjlisttrnfile MyServer -gid d138000100000000000000000000fefb57e648000000
0000000001
```

For details on the `cjlisttrnfile` command, see *cjlisttrnfile (display transaction information for stopped J2EE server)* in the *uCosminexus Application Server Command Reference Guide*.

Appendix

A. Migrating from Server Plug-in

This appendix describes how to use alternative means such as the management portal, standard DD, and `cosminexus.xml` for performing the operations that were performed in earlier versions by using Server Plug-in.

This section describes the alternative means for the following Server Plug-in operations:

- Operation management of a logical server
  Describes operations such as starting and stopping a logical server and displaying the operation status.
- Operations of a resource adapter
  Describes operations such as importing exporting, and deploying a resource adapter.
- Operations of a J2EE application
  Describes operations such as importing, exporting, and starting and stopping a J2EE application.
- Settings of a resource adapter included in a J2EE application
  Describes operations such as importing, starting, and stopping a J2EE application including a resource adapter.
- Property settings of a J2EE application
  Describes the following items:
  - Property settings of an Enterprise Bean
  - Property settings of a servlet and JSP
  - Common property settings of a J2EE application

Note that in the tables below, *uCosminexus Application Server* has been omitted from the manual names described in the *Reference manual* column describing the reference locations. Each operation is described in the following sections.

A.1 Operation management of a logical server

The following table describes the means for the operation management of a logical server that was previously performed by using Server Plug-in.

Table A–1: Methods for operation management of a logical server and their reference locations

Server Plug-in operation		Alternative means	Reference manual	Reference location
Operation overview	Operation details			
Starting a logical server	Setting the start option	Management portal	*Management Portal User Guide*	*10.8.2*
	Starting logical servers in the domain unit, in a batch	Management portal	*Management Portal User Guide*	*11.3.2*
	Starting logical servers in the host unit, in a batch	Management portal	*Management Portal User Guide*	*11.2.2*
	Starting a logical server individually	Management portal	*Management Portal User Guide*	*11.4.2*, *11.5.2*, *11.6.2*, *11.7.2*, *11.8.2*, *11.9.2*, *11.10.2*, *11.10.6*, *11.11.2*, *11.12.2*, *11.12.6*, *11.13.2*
	Starting J2EE server without starting an application	Management portal	*Management Portal User Guide*	*11.9.2*

Server Plug-in operation		Alternative means	Reference manual	Reference location
Operation overview	Operation details			
Stopping a logical server	Stopping logical servers in the domain unit, in a batch	Management portal	*Management Portal User Guide*	*11.3.3*
	Stopping logical servers in the host unit, in a batch	Management portal	*Management Portal User Guide*	*11.2.3*
	Stopping a logical server individually	Management portal	*Management Portal User Guide*	*11.4.3*, *11.5.3*, *11.6.3*, *11.7.3*, *11.8.3*, *11.9.3*, *11.10.3*, *11.10.7*, *11.11.3*, *11.12.3*, *11.12.7*, *11.13.3*
	Stopping a logical J2EE server forcibly	Management portal	*Management Portal User Guide*	*11.10.7*
Displaying the operation status of the logical server	Operation status of all logical servers within the operation management domain	Management portal	*Management Portal User Guide*	*11.3.1*

A.2 Operations of a resource adapter

The following table describes the means for the operations of resource adapters that were previously performed by using Server Plug-in.

Table A–2: Operation methods of a resource adapter and their reference locations

Operation by Server Plug-in	Alternative means	Reference manual	Reference location
Importing a resource adapter	Management portal	*Management Portal User Guide*	*12.4.4*
Defining the properties of a resource adapter	Management portal	*Management Portal User Guide*	*12.4.5*
Deploying a resource adapter	None (executed automatically after import).	--	--
Performing the connection test of a resource adapter	Management portal	*Management Portal User Guide*	*12.4.3*
Starting a resource adapter	Management portal	*Management Portal User Guide*	*12.4.1*
Stopping a resource adapter	Management portal	*Management Portal User Guide*	*12.4.2*
Deleting a resource adapter	Management portal	*Management Portal User Guide*	*12.4.6*
List of operating states of a resource adapter	Management portal	*Management Portal User Guide*	*12.4.1*
Exporting a resource adapter	Command (`cjexportrar`)	*Command Reference Guide*	*cjexportrar* (export of a resource adapter)
Importing and exporting the HITACHI Connector property#	Management portal	*Management Portal User Guide*	*12.4.5*

Legend:

--: Not applicable.

#

You can specify and reference the HITACHI Connector property in the Edit Connector Property File of the Resource Adapter window in the management portal.

A.3 Operations of a J2EE application

The following table describes the means for the operations of J2EE applications that were previously performed by using Server Plug-in.

Table A–3: Operation methods of a J2EE application and their references

Operation by Server Plug-in	Alternative means	Reference manual	Reference location
Importing J2EE applications (archive format)	Management portal	*Management Portal User Guide*	*12.3.3*
Importing J2EE applications (exploded archive format)	Management portal	*Management Portal User Guide*	*12.3.4*
Starting J2EE applications	Management portal	*Management Portal User Guide*	*12.3.1*
Stopping J2EE applications normally	Management portal	*Management Portal User Guide*	*12.3.2*
Stopping J2EE applications forcibly	Command (`cjstopapp`)	*Command Reference Guide*	*cjstopapp* (stopping J2EE applications)
Referencing the list of J2EE applications	Management portal	*Management Portal User Guide*	*12.3.1*
Deleting J2EE applications	Management portal	*Management Portal User Guide*	*12.3.5*
Replacing J2EE applications (archive format)	Management portal	*Management Portal User Guide*	*4.3*
Replacing J2EE applications (exploded archive format)	Management portal	*Management Portal User Guide*	*4.3*
Changing a J2EE application name	Command (`cjrenameapp`)	*Command Reference Guide*	*cjrenameapp* (Changing an application name)
Acquiring RMI-IIOP stubs and interfaces	Command (`cjgetstubsjar`)	*Command Reference Guide*	*cjgetstubsjar* (Acquiring RMI-IIOP stubs and interfaces of an application)
Importing the application integrated property	Command (`cjsetappprop`)#	*Command Reference Guide*	*cjsetappprop* (Setting the properties of an application)
Exporting the application integrated property	Command (`cjgetappprop`)#	*Command Reference Guide*	*cjgetappprop* (Acquiring the properties of an application)
Exporting J2EE applications	Command (`cjexportapp`)	*Command Reference Guide*	*cjexportapp* (Exporting J2EE applications)

#

We recommend that you use `cosminexus.xml` to manage the application information. For details on J2EE applications including `cosminexus.xml`, see *13. Managing the Properties of an Application* in the *uCosminexus Application Server Common Container Functionality Guide*.

A.4 Settings of a resource adapter included in a J2EE application

The following table describes the means for the settings of resource adapters included in J2EE applications that were previously specified by using Server Plug-in.

Table A–4: Operation methods for setting a resource adapter and their reference locations

Operation by Server Plug-in	Alternative means	Reference manual	Reference location
Importing J2EE applications including a resource adapter	Management portal	*Management Portal User Guide*	*12.3.3*
Defining properties of a resource adapter	`cosminexus.xml` (such as the `rar` tag)	*Application and Resource Definition Reference Guide*	*2.2*
Performing the connection test of a resource adapter	Command (`cjtestres`)	*Command Reference Guide*	*cjtestres* (Connection test of a resource)
Starting J2EE applications including a resource adapter	Management portal	*Management Portal User Guide*	*12.3.1*
Stopping J2EE applications including a resource adapter	Management portal	*Management Portal User Guide*	*12.3.2*
Importing and exporting the HITACHI Connector property	Command (`cjgetappprop`)	*Command Reference Guide*	*cjgetappprop* (Acquiring the properties of an application)
	Command (`cjsetappprop`)	*Command Reference Guide*	*cjsetappprop* (Setting the properties of an application)

A.5 Property settings of a J2EE application

The following items are described below:

- Property settings of an Enterprise Bean
- Property settings of a servlet and JSP
- Common property settings of a J2EE application

(1) Property settings of an Enterprise Bean

The following table describes the means for the Enterprise Bean property settings that were previously specified by using Server Plug-in.

Table A–5: Operation methods for property settings of an Enterprise Bean and their reference locations

Operation by Server Plug-in	Alternative means	Reference location
Defining other Enterprise Bean references (`ejb-ref`)	DD (`ejb-jar.xml`)	*9.3.1*
Defining the reference of a resource adapter (`resource-ref`)	`cosminexus.xml`, DD (`ejb-jar.xml`)	*9.3.3*
Defining the reference of a resource environment (`resource-env-ref`)	cosminexus.xml, DD (`ejb-jar.xml`)	*9.3.4*
Defining the message reference of a Message-driven Bean (`message-ref`)	`cosminexus.xml`	*9.4*
Defining a transaction attribute (`trans-attribute`)	DD (`ejb-jar.xml`)	*9.5*
Setting properties when executing a Stateful Session Bean	`cosminexus.xml`	*9.10.1*
Setting properties when executing a Stateless Session Bean	`cosminexus.xml`	*9.10.2*
Setting properties when executing an Entity Bean	`cosminexus.xml`	*9.10.3*
Setting properties when executing a Message-driven Bean	`cosminexus.xml`	*9.10.4*

(2) Property settings of a servlet and JSP

The following table describes the means for the settings of servlet and JSP properties that were previously specified by using Server Plug-in.

Table A–6: Operation methods for property settings of a servlet and JSP

Operation by Server Plug-in	Alternative means	Reference location
Defining the reference of an Enterprise Bean (`ejb-ref`)	DD (`web.xml`)	*9.7.1*
Defining the reference of a resource adapter (`resource-ref`)	`cosminexus.xml`, DD (`web.xml`)	*9.7.3*
Defining the reference of a resource environment (`resource-env-ref`)	`cosminexus.xml`, DD (`web.xml`)	*9.7.4*
Defining the mapping of a servlet and JSP (`url-pattern`)	DD (`web.xml`)	*9.8*
Defining the mapping of a filter (`filter-mapping`)	DD (`web.xml`)	*9.9.2*
Defining the context root of a J2EE application (`context-root`)	DD (`application.xml`)	*9.11.1*
Defining control for the number of concurrently executing threads in a Web application unit (`thread-control`)	`cosminexus.xml`	*9.11.2*
Defining control for the number of concurrently executing threads in a URL group unit (`urlgroup-thread-control`)	`cosminexus.xml`	*9.11.3*
Defining the monitoring of the number of pending requests in a URL group unit (`urlgroup-thread-control` tag - `stats-monitor` tag - `waiting-request-count` tag)	`cosminexus.xml`	*9.11.4*
Setting the default character encoding (`http-request` tag - `encoding` tag, `http-response` tag - `encoding` tag, `jsp` tag - `page-encoding` tag)	`cosminexus.xml`	*9.12*
Setting the error reporting for a servlet and JSP	`cosminexus.xml`	*9.16*

(3) Common property settings of a J2EE application

The following table describes the means for the common J2EE property settings that were previously specified by using Server Plug-in.

Table A–7: Operation methods for common property settings of a J2EE application

Operation by Server Plug-in		Alternative means	Reference location
Operation overview	Operation details		
CTM scheduling	Scheduling of a J2EE application unit	`cosminexus.xml`	*9.14.1*
	Scheduling of a Stateless Session Bean unit	`cosminexus.xml`	*9.14.2*
Setting the startup order	Setting the startup order of a J2EE application	`cosminexus.xml`	*9.15.1*
	Setting the startup order of an Enterprise Bean	`cosminexus.xml`	*9.15.2*
	Setting the startup order of a servlet and JSP	`cosminexus.xml`	*9.15.3*

Index