IBM WebSphere Adapters
Version 7 Release 5 Fix Pack 2 (7.5.0.2)

IBM WebSphere Adapter for JD Edwards EnterpriseOne User Guide Version 7 Release 5 Fix Pack 2 (7.5.0.2)

IBM WebSphere Adapters
Version 7 Release 5 Fix Pack 2 (7.5.0.2)

IBM WebSphere Adapter for JD Edwards EnterpriseOne User Guide Version 7 Release 5 Fix Pack 2 (7.5.0.2)

Note
Before using this information and the product it supports, read the information in "Notices" on page 115.

May 2012

This edition applies to version 7, Release 5, Fix Pack 2 (7.5.0.2) of IBM WebSphere Adapter for JD Edwards EnterpriseOne and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email mailto://doc-comments@us.ibm.com. We look forward to hearing from you.

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

Contents

Chapter 1. Overview of WebSphere Adapter for JD Edwards EnterpriseOne

With WebSphere® Adapter for JD Edwards EnterpriseOne, you can create integrated processes that include the exchange of information with a JD Edwards EnterpriseOne server, without special coding.

The adapter provides a standard interface that eliminates the need for the component to understand the lower-level implementation details or data structures of the application. Using the adapter, a component (the program or piece of code that performs a specific business function) can send requests to the JD Edwards EnterpriseOne server (for example, to query a customer record in a JD Edwards EnterpriseOne table or to update an order document).

WebSphere Adapter for JD Edwards EnterpriseOne complies with the Java Platform, Enterprise Edition (JEE) Connector Architecture (JCA). JCA standardizes the way application components, application servers, and enterprise information systems, such as a JD Edwards EnterpriseOne server, interact with each other. WebSphere Adapter for JD Edwards EnterpriseOne makes it possible for JCA-compliant application servers to connect to and interact with the JD Edwards EnterpriseOne server. Clients running on the JCA-compliant server can then communicate with the JD Edwards EnterpriseOne server in a standard way (using business objects or JavaBeans).

The following example assumes you are setting up an adapter using Rational® Application Developer for WebSphere Software and deploying the module that contains the adapter to WebSphere Application Server.

Suppose a medium-sized retail company uses JD Edwards EnterpriseOne to coordinate most of its business operations. JD Edwards EnterpriseOne includes a business function that can return a real-time list of inventory items for its 100 stores located across the United States. An application component might be able to use this business function as part of an overall business process. For example, an employee of a retail company can access the real-time list of available inventory items, thus providing correct, real-time information to a customer.

Hardware and software requirements

The hardware and software requirements for WebSphere Adapters are provided on the IBM® Support website.

To view hardware and software requirements for WebSphere Adapters, see http://www.ibm.com/support/docview.wss?uid=swg27006249.

Additional information

The following links provide additional information you might need to configure and deploy your adapter:

- The compatibility matrix for WebSphere Business Integration Adapters and WebSphere Adapters identifies the supported versions of required software for your adapter. To view this document, go to the WebSphere Adapters support

page: http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.

- Technotes for WebSphere Adapters provide workaround and additional information that are not included in the product documentation. To view the technotes for your adapter, go to the following Web page, select the name of your adapter from the **Product category** list, and click the search icon: http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm.

Technical overview of the Adapter for JD Edwards EnterpriseOne

IBM WebSphere Adapter for JD Edwards EnterpriseOne provides a way for applications to interact with data on JD Edwards EnterpriseOne servers. Outbound processing, which is the processing of requests from an application through the adapter to the JD Edwards EnterpriseOne server, is supported.

The adapter processes requests using one of two types of business objects: business functions and XML Lists. A business function is a business object container that can contain one or many business objects which can be processed as a single transaction. An XML List is a single business objects that can query a table and return multiple records.

You create business objects by using the J2C Bean wizard, a tool launched from Rational Application Developer for WebSphere Software. The business objects generated by the J2C Bean wizard have predefined business object definitions. If you want to add or remove functionality from a generated business object, you can use the business object editor, another tool launched from Rational Application Developer for WebSphere Software, to change the properties of the generated business object definition. You can also change or set configuration properties for business objects in the administration console of Rational Application Developer for WebSphere Software.

Outbound processing

Adapter for JD Edwards EnterpriseOne supports synchronous outbound request processing. This means that when the adapter receives a request, in the form of a business object, from the module, it processes the request and returns the result, when applicable, in a business object.

WebSphere Adapter for JD Edwards EnterpriseOne business applications supports synchronous outbound processing. This implies that the component sends a request in the form of a WebSphere business object hierarchy to the adapter, the adapter processes the request and returns a WebSphere business object hierarchy that represents the result of the operation.

When the adapter receives a WebSphere business object hierarchy, the adapter processes it as follows:

1. Extracts metadata from the WebSphere business object hierarchy.
2. Identifies the appropriate JD Edwards EnterpriseOne objects to access (for example, JD Edwards EnterpriseOne business objects and business components, or JD Edwards EnterpriseOne business function, or JD Edwards EnterpriseOne XML List) depending on the objects against which the artifacts were generated.
3. Extracts the outbound operation to perform from the WebSphere business object hierarchy.

4. Retrieves the data for XML List, or performs the corresponding business function method, after accessing the required JD Edwards EnterpriseOne objects.
5. Populates that JD Edwards EnterpriseOne object (business function or XML List hierarchy) with data from the hierarchy of WebSphere business objects, if there are updates (Create, Update, Delete), the adapter.

Supported outbound operations

When the adapter receives a request, it processes the request using the JD Edwards EnterpriseOne Dynamic Java connector to invoke either a business function or an XML List.

Business functions support the following types of operations:

- Create
- Delete
- Execute
- Retrieve
- Update

XML Lists support the following operation:

- retrieveAll

Table 1.

Operations	Description
Create	Creates the business object.
Delete	Deletes the business object and its children. Because the adapter supports only logical deletes, objects are marked as deleted but not removed.
Exists	Checks for the existence of incoming business objects.
Retrieve	Retrieves the JD Edwards EnterpriseOne component and maps component data onto the business object.
RetrieveAll	Retrieves multiple instances of the JD Edwards EnterpriseOne component and maps component data onto the business object.
Update	Updates the corresponding JD Edwards EnterpriseOne component with the incoming business object.

Inbound processing

WebSphere Adapter for JD Edwards EnterpriseOne supports asynchronous inbound processing. This means that the adapter polls theJD Edwards EnterpriseOne server at specified intervals for events. When the adapter receives an event, it converts the event data into a business object and sends the business object to the component.

WebSphere Adapter for JD Edwards EnterpriseOne supports real-time events. A real-time event is a business transaction that provides information from the JD Edwards EnterpriseOne server that can be used to interoperate with a third-party system. Real-time events can be generated wherever business functions run, such as HTML, WIN32, and enterprise servers. Real-time events are useful for producing notifications in real-time. The adapter supports both single and container real-time events.

When the adapter gets a real-time event from the JD Edwards EnterpriseOne transaction server by invoking JD Edwards EnterpriseOne Dynamic Java Connector API, it parses the content of this real-time event and converts it into a business object. The adapter then sends the business object to the event endpoints. For example, if a company is updated, the JD Edwards EnterpriseOne server captures this change immediately, and one real-time event is generated by JD Edwards EnterpriseOne transaction server. The adapter then communicates with the JD Edwards EnterpriseOne transaction server, retrieves this real-time event, and processes it. After converting it into a business object, the adapter delivers this business object to the event endpoint.

WebSphere Adapter for JD Edwards EnterpriseOne processes events as follows:

1. Invokes JD Edwards EnterpriseOne Dynamic Java Connector API to get a real-time event.
2. Parses the content of this real-time event.
3. Populates the associated business object with the values it retrieves from the payload of this real-time event.
4. Sends the generated business object to each registered application.

Note: You must configure the JD Edwards EnterpriseOne to support real-time events before inbound processing operation.

Event persistence

The adapter supports event persistence for inbound processing in case of abrupt termination. Event persistence (or assured-once delivery) is a way to make sure that events are delivered once, and only once, to the endpoint in the case of a failure. During event processing, the adapter persists the event state in an event store located at the data source.

You must set up this data source using WebSphere Application Server before you can create the event store. To use the recovery feature provided by WebSphere Application Server, you set the `AssuredOnceDelivery` property in the activation specification to `True`. This recovery feature is set to `True` by default.

The adapter also provides for event persistence using an in-memory representation of the event store. When you use this feature, you need not create a JNDI data source or an external event store, and event processing is faster. However, with this feature there is no support for event recovery. In the case of server failure, the in-memory event stores are lost.

To prevent the loss of events in the case of server failure, the recommended approach is to use the database event store. To use the in-memory event persistence capability of the adapter, you must not set the `EP_DataSource_JNDIName` property.

When a failed event occurs and the file cannot be written to disk the JDE adapter will print the content of this failed event to a JDE trace file, to avoid event loss. The content will be between two "##########", and the trace level is INFO. The content can be copied into a new file and saved with the event ID as the file name in the directory "FailedEventFolder". When the status of the failed event record is changed from -1 to 0 in the event table the event will be handled again.

Event store

The event store is a persistent cache where event records are saved until the polling adapter can process them. The adapter uses event stores to keep track of

inbound events as they make their way through the system. Each time a real-time event is received, the adapter updates the status of the event in an event store. The status of each event is continually updated by the adapter for recovery purposes until the events are delivered to the endpoint.

If the adapter detects that there is no event store for the inbound module in the WebSphere Application Server, it automatically creates one when the application is deployed to the runtime. Each event store created by the adapter is associated with a specific inbound module. The adapter does not support multiple adapter modules pointing to the same event store.

When the adapter polls the JD Edwards EnterpriseOne transaction server and receives a real-time event, it creates an entry in the event store for each event that matches the search criteria specified in the activation specification properties. The adapter records the status of each new entry as NEW.

If a real-time event is successfully delivered, the corresponding event store entries are deleted. For failed events, the entries remain in the event store.

Assured-once delivery

The JD Edwards EnterpriseOne transaction server provides guaranteed event delivery quality of service. This means that all the real-time events that the adapter subscribes to are delivered to the adapter without any loss of events. Because it is possible for the JD Edwards EnterpriseOne transaction server to send duplicate real-time events to the adapter, the adapter provides assured-once event delivery. This means that each event is delivered once and only once. To enable assured-once delivery, you must set the `AssuredOnceDelivery` activation specification property to `True`.

Note: When you set the `AssuredOnceDelivery` activation specification property to `True`, you must set the `AutoAcknowledge` activation specification property to `False`. Otherwise, the assured-once delivery feature will not work.

When you set the `AssuredOnceDelivery` activation specification property to `True`, the adapter stores an XID (transaction ID) value for each event in the event store.

When a real-time event is obtained, it is processed as follows:

1. The XID value for the event is updated in the event store.
2. The event is delivered to its corresponding endpoint.
3. The event entry is deleted from the event store.
4. An acknowledgement is issued to the JD Edwards EnterpriseOne transaction server.

Monitoring inbound events

The adapter supports monitoring inbound events from the JD Edwards EnterpriseOne server, in addition to the other events you are monitoring using Business Monitor or WebSphere Business Events.

Monitoring inbound events using IBM Business Monitor:

You can use Rational Application Developer for WebSphere Software and the adapter to send inbound events to WebSphere Application Server Common Event Infrastructure (CEI), where they are accessible to Business Monitor.

When you select the option to monitor inbound events in the Rational Application Developer for WebSphere Software J2C Bean wizard, the required artifacts are generated to monitor inbound events. These artifacts include the message-driven J2C bean, as well as the interface, Service Bean, interceptor class, helper class, and the event schemas that are required to create a monitor model. You can then deploy the resulting adapter inbound event monitoring application containing the message-driven bean (the adapter application) to a server, either a Business Monitor server or a remote server. The message-driven bean invokes the stateless session bean that makes the events accessible to the client. More importantly, it also listens for events coming in from the JD Edwards EnterpriseOne server (inbound events) and uses the interceptor to set up the intercepted inbound events as Common Base Events (CBE). Then, it posts these Common Base Events into a designated Java Message Service (JMS) destination - Common Event Infrastructure queue, where they are accessible to Business Monitor for further processing.

Important: Inbound event monitoring is available to your application only if you have Business Monitor installed in your environment. For more information about installing Business Monitor, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v8r0mx/topic/com.ibm.wbpm.mon.imuc.doc/inst/intro.html. For more information about the software requirements and configuration, see http://www-304.ibm.com/support/docview.wss?uid=swg27008414.

For enabling inbound event monitoring function, perform the following tasks:

1. Launch the J2C Bean wizard (see "Launching the J2C Bean wizard" on page 19)
2. Add the software dependencies (see "Configuring the connector dependencies " on page 19)
3. Edit the dependency files (see "Editing connector dependency files" on page 22)
4. Set the connection properties in J2C Bean wizard (see "Setting connection properties for the J2C Bean wizard" on page 23)
5. Configuring the module for inbound processing (see "Configuring the module for inbound processing" on page 37)
6. Setting deployment properties and generating artifacts (see Setting deployment properties and generating artifacts)

Related References

For a sample on enabling inbound event monitoring for Business Monitor, see http://publib.boulder.ibm.com/infocenter/radhelp/v8r5/topic/com.ibm.j2c.doc/topics/tcreatinginboundapps.html.

For information about how to disable the event monitor function, see http://publib.boulder.ibm.com/infocenter/radhelp/v8r5/topic/com.ibm.j2c.doc/topics/tdisablingwbe.html.

For a sample end-to-end scenario on publishing events to the Business Monitor, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v8r0mx/topic/com.ibm.wbpm.mon.doc/scen/eis.html.

Monitoring inbound events using WebSphere Business Events:

You can use Rational Application Developer for WebSphere Software and the adapter to send inbound events to WebSphere Application Server JMS topic, where they are accessible to WebSphere Business Events.

Note: You cannot create a JMS connection to the remote server when the same connection factory name is duplicated. For more information, see http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.pmc.doc/tasks/tjn0033_.html

When you select the option to monitor inbound events in the Rational Application Developer for WebSphere Software J2C Bean wizard, the required artifacts are generated to monitor inbound events. These artifacts include the message-driven J2C bean, as well as the interface, Service Bean, interceptor class, helper class, and the event schemas that are required to create a monitor model. You can then deploy the resulting adapter inbound event monitoring application containing the message-driven bean (the adapter application) to a server. The message-driven bean invokes the stateless session bean that makes the events accessible to the client. More importantly, it also listens for events coming in from the JD Edwards EnterpriseOne server (inbound events) and uses the interceptor to set up the intercepted inbound events as Common Base Events (CBE). Then, it posts these Common Base Events into a designated Java Message Service (JMS) destination - JMS topic, where they are accessible to WebSphere Business Events for further processing.

Important: Inbound event monitoring is available to your application only if you have WebSphere Business Events installed in your environment. For information about installing WebSphere Business Events, see http://publib.boulder.ibm.com/infocenter/wbevents/v6r2m1/index.jsp?topic=/com.ibm.wbe.install.doc/doc/install.html. The WebSphere Business Events works with WebSphere Application Server version 6.1; it is not supported in WebSphere Application Server version 7.0. For more information about the software requirements and configuration, see http://www.ibm.com/software/integration/wbe/requirements/.

For enabling inbound event monitoring function, perform the following tasks:

1. Launch the J2C Bean wizard (see “Launching the J2C Bean wizard” on page 19)
2. Add the software dependencies (see “Configuring the connector dependencies ” on page 19)
3. Edit the dependency files (see “Editing connector dependency files” on page 22)
4. Set the connection properties in J2C Bean wizard (see “Setting connection properties for the J2C Bean wizard” on page 23)
5. Configuring the module for inbound processing (see “Configuring the module for inbound processing” on page 37)
6. Setting deployment properties and generating artifacts (see Setting deployment properties and generating artifacts)
7. From the inbound session bean, generate the eventBOTypeMapping.xml and eventBOTypeMapping.xsd files. The eventMapping file provides mapping information between an event and the business object schema, which the WebSphere Business Event requires to monitor an event. To generate the eventBOTypeMapping.xml and eventBOTypeMapping.xsd files:
   a. Right-click your session bean.
   b. Select **Source** > **Generate Event Mapping**.

   The EventMapping files are generated in the same folder as your business object schema files.

Related Reference

For integrating WebSphere Business Events with WebSphere Application Adapters, see http://publib.boulder.ibm.com/infocenter/wbevents/v6r2m1/topic/com.ibm.wbe.integrating.doc/doc/integratingusingwebsphereadapters.html.

Business objects

A business object is a structure that consists of data, the action to be performed on the data, and additional instructions, if any, for processing the data. The data can represent either a business entity, such as an invoice or an employee record, or unstructured text. The adapter uses business objects to send data to, or obtain data from the JD Edwards EnterpriseOne server.

How the adapter uses business objects

The adapter uses the JD Edwards EnterpriseOne Dynamic Java Connector APIs to communicate with the JD Edwards EnterpriseOne application. The adapter exchanges information with JD Edwards EnterpriseOne through business functions, XML List calls and the real-time event mechanism.

How business objects are created

You create business objects by using the J2C Bean wizard, which is launched from Rational Application Developer for WebSphere Software. The J2C Bean wizard connects to the application, discovers data structures in the application, and generates business objects to represent them. It also generates other artifacts needed by the adapter.

Business object structure

The adapter supports processing of hierarchical business objects. A container business object representing a JD Edwards EnterpriseOne operation is a wrapper object that contains single or multiple child business function objects, also called simple business function objects. Each business function object represents a specific function call in the JD Edwards EnterpriseOne application.

Service Bean

The business data exchanged between the client application and the resource adapter is represented as Service Bean. The metadata describing the business data is defined as business objects and represented as the XSD schemas. The Service Bean is generated from these XSDs and is the realization of the business objects.

A Service Bean is a structure that consists of data and, in some cases, metadata with additional instructions, for processing the data. It is a generated, hierarchical, Java objects implementing Record interface. The data can represent a business entity, such as an invoice or an employee record.

You create Service Bean by using the J2C Bean wizard, launched from Rational Application Developer for WebSphere Software connector tools. The wizard connects to the JD Edwards EnterpriseOne, discovers data structures in the EIS, and generates Service Bean to represent them. The adapter supports records that are hierarchically structured. Information about the processed object is stored in the application-specific information for the object and each of its attributes.

The J2C Bean wizard

The J2C Bean wizard is a tool you use to configure your adapter before deploying it to WebSphere Application Server. The J2C Bean wizard establishes a connection to the JD Edwards EnterpriseOne server, discovers services (based on search criteria you provide), and generates business objects and interfaces based on the services discovered.

Standards Compliance

This product is compliant with several government and industry standards, including accessibility standards and Internet Protocol standards.

Accessibility

Administration

The run time administrative console is the primary interface for deployment and administration of enterprise applications. These consoles are displayed within a standard web browser. By using an accessible Web browser, such as Microsoft Internet Explorer or Netscape Browser, you are able to:

- Use screen-reader software and a digital speech synthesizer to hear what is displayed on the screen
- Use voice recognition software, such as IBM via Voice, to enter data and to navigate the user interface
- Operate features by using the keyboard instead of the mouse

You can configure and use product features by using standard text editors and scripts or command line interfaces instead of the graphical interfaces that are provided. When appropriate, the documentation for specific product features contains additional information about the accessibility of the features.

J2C Bean wizard

The J2C Bean wizard is the primary component used to create application accessing EIS systems. This wizard is implemented as an Eclipse plug-in that is available through Rational Application Developer for WebSphere Software is fully accessible.

Keyboard navigation

This product uses standard Microsoft Windows® navigation keys.

IBM and accessibility

See the IBM Accessibility Center website http://www.ibm.com/able/ for more information about the commitment that IBM has to accessibility.

Internet Protocol, Version 6 (IPv6)

WebSphere Application Server, version 6.1.0 and later and its JavaMail component support dual-stack Internet Protocol Version 6.0 (IPv6). For more information about this compatibility in WebSphere Application Server, see IPv6 support in the WebSphere Application Server information center. For more information about IPv6, see http://www.ipv6.org.

Log and Trace Analyzer

The adapter creates log and trace files that can be viewed with the Log and Trace Analyzer.

The Log and Trace Analyzer can filter log and trace files to isolate the messages and trace information for the adapter. It can also highlight the adapter's messages and trace information in the log viewer.

The adapter's component ID for filtering and highlighting is a string composed of the characters `JDERA` plus the value of the adapter ID property. For example, if the adapter ID property is set to `001`, the component ID is `JDERA001`.

If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for JD Edwards EnterpriseOne to `001` and `002`. The component IDs for those instances, `JDERA001` and `JDERA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `JDERAInstance`.

For outbound processing, the adapter ID property is located in both the resource adapter and managed connection factory property groups. If you update the adapter ID property after using the J2C Bean wizard to configure the adapter for outbound processing, be sure to set the resource adapter and managed connection factory properties consistently. It prevents inconsistent marking of the log and trace entries. For inbound processing, the adapter ID property is located only in the resource adapter properties, so this consideration does not apply.

For more information, see the “Adapter ID (AdapterID)” on page 87 property.

Chapter 2. Planning for adapter implementation

To implement the application using IBM® WebSphere® Adapter for JD Edwards EnterpriseOne, you must plan for request processing and consider security and performance requirements.

Before you begin

Before you begin to set up and use the adapter, you should possess a thorough understanding of the capabilities and requirements of the integration development tools and runtime environment you will use, and the JD Edwards EnterpriseOne environment where you will build and use the solution.

To configure and use WebSphere Adapter for JD Edwards EnterpriseOne, you should understand and have experience with the following concepts, tools, and tasks:

- The business requirements of the solution you are building.
- The capabilities provided by the integration development tools you will use to build the solution. You should know how to use these tools to create modules, test components, and complete other integration tasks.
- The capabilities and requirements of the runtime environment you will use for the integration solution. You should know how to configure and administer the host server and how to use the administrative console to set and modify property definitions, configure connections, and manage events.

Note: When you deploy the JDE adapter in IBM Business Process Manager that is running on a Unix platform, you need to run the following configuration file in IBM Business Process Manager, else an exception error is thrown by IBM Business Process Manager.

The configuration file, jaxp.properties.sample, can be located in the lib subdirectory of the JRE of IBM Business Process Manager. The javax.xml.parsers.SAXParserFactory, javax.xml.parsers.DocumentBuilderFactory, javax.xml.xpath.XPathFactory, and javax.xml.transform.TransformerFactory properties are defined in the jaxp.properties.sample file, but are commented out. You need to uncomment the javax.xml.parsers.SAXParserFactory, javax.xml.parsers.DocumentBuilderFactory, javax.xml.xpath.XPathFactory and javax.xml.transform.TransformerFactory properties and rename the file from jaxp.properties.sample to jaxp.properties.

Support for protecting sensitive user data in log and trace files

You can configure the adapter to prevent sensitive or confidential data, in the log and trace files, from being viewed by users without authorization.

Log and trace files for the adapter can contain data from your JD Edwards EnterpriseOne server, which might contain sensitive or confidential information. Sometimes these files might be seen by individuals without authorization to view sensitive data. For example, a support specialist must use the log and trace files to troubleshoot a problem.

To protect the data in such situations, the adapter lets you specify whether you want to prevent confidential user data from displaying in the adapter log and trace files. You can select this option in the J2C Bean wizard or change the HideConfidentialTrace property. When this property is enabled, the adapter replaces the sensitive data with XXX's.

See “Managed connection factory properties” on page 90 for information about this optional property.

The following types of information are considered potentially sensitive data and are disguised:

- The contents of a business object
- The contents of the object key of the event record
- User name, Password, Environment, and Role
- The URL used to connect to the JD Edwards EnterpriseOne server

The following types of information are not considered user data and are not hidden:

- The contents of the event record that are not part of the event record object key, for example, the XID, event ID, business object name, and event status
- Business object schemas
- Transaction IDs
- Call sequences

Security

The adapter uses the J2C authentication data entry, or the authentication alias feature of Java 2 security to provide secure user name and password authentication. For more information about security features, see the documentation for WebSphere Application Server.The adapter is Java 2 security enabled and features user name and password authentication. In addition, you can configure additional security permissions by altering the application server's “WAS.policy” file and storing it in the meta-inf folder. For more details on configuring security details, see the security documentation for WebSphere Application Server.

User authentication

The adapter supports several methods for supplying the user name and password that are needed to connect to the JD Edwards EnterpriseOne server. By understanding the features and limitations of each method, you can pick a method that provides the appropriate level of security and convenience for your application.

To integrate an adapter into your application, a user name and password are needed at the following times:

- When the J2C Bean wizard connects to JD Edwards EnterpriseOne server to extract, or *discover*, information about the objects and services that you can access with the adapter.
- At run time on WebSphere Application Server, when the adapter connects to JD Edwards EnterpriseOne server to process outbound requests.

Authentication in the wizard

The J2C Bean wizard prompts for the connection information for the discovery process, and then reuses it as the default values of the adapter properties that specify the connection information used at run time. You can use a different user name and password while running the wizard than you use when the application is deployed to the server. You can even connect to a different JD Edwards EnterpriseOne server, although the schema name must be the same in both databases. For example, while developing and integrating an application that uses WebSphere Adapter for JD Edwards EnterpriseOne, you might not use the production database; using a test database with the same data format but fewer, simulated records lets you develop and integrate the application without impacting the performance of a production database and without encountering restrictions caused by the privacy requirements for customer data.

The wizard uses the user name and password that you specify for the discovery process only during the discovery process; they are not accessible after the wizard is completed.

Authentication at run time

At run time, the adapter needs to provide the user name and password to connect to the JD Edwards EnterpriseOne server. To connect without user intervention, the adapter must access a saved copy of the user information. In a server environment, there are several methods for saving user information. You can configure the adapter to get your user information, through any of the following methods:

- Adapter properties
- J2C authentication alias

Saving the user name and password in adapter properties is a direct way to provide this information at run time. You provide this user name and password when you use the J2C Bean wizard to configure your module. Although directly specifying the user name and password seems the most straightforward method, it has important limitations. Adapter properties are not encrypted; the password is stored as clear text in fields that are accessible to others on the server. Also, when the password changes, you must update the password in all instances of the adapter that access that JD Edwards EnterpriseOne server. This includes the adapters embedded in application EAR files as well as adapters that are separately installed on the server.

Using a J2C authentication data entry, or authentication alias, created with the Java Authentication and Authorization Service (JAAS) feature of Java 2 security is a robust, secure way to deploy applications. An administrator creates the authentication alias that is used by one or more applications that need to access a system. The user name and password must be known only to that administrator, who can change the password in a single place, when a change is required.

Deployment options

There are two ways to deploy the adapter. You can either embed it as part of the deployed application, or you can deploy it as a stand-alone RAR file. The requirements of your environment affect the type of deployment option you choose.

The following are the deployment options:

- When you deploy the adapter as an embedded component, the adapter is included within an enterprise application archive (EAR) file and is available only to the application in the EAR file.
- When you deploy the adapter as a stand-alone component, the adapter is represented by a stand-alone resource adapter archive (RAR) file. When it is deployed, it is available to all applications deployed in the server instance.
- **With module for use by single application**: With the adapter files embedded in the module, you can deploy the module to any application server. Use an embedded adapter when you have a single module using the adapter or if multiple modules need to run different versions of the adapter. Using an embedded adapter enables you to upgrade the adapter in a single module without the risk of destabilizing other modules by changing their adapter version.
- **On server for use by multiple applications**: If you do not include the adapter files in a module, you must install them as a stand-alone adapter on each application server where you want to run the module. Use a stand-alone adapter when multiple modules can use the same version of the adapter and you want to administer the adapter in a central location. A stand-alone adapter can also reduce the resources required by running a single adapter instance for multiple modules.

While creating the project for your application using Rational Application Developer for WebSphere Software, you can choose how to package the adapter [either bundled with the (EAR) file or as a stand-alone (RAR) file]. Your choice affects how the adapter is used in the run time environment, as well as how the properties for the adapter are displayed on the administrative console.

Choosing either to embed an adapter with your application or to deploy the adapter as a stand-alone module depends on how you want to administer the adapter. If you want a single copy of the adapter and do not care about disruption to multiple applications when you upgrade the adapter, then you would be more likely to deploy the adapter as a stand-alone module.

If you plan to run multiple versions, and if you care more about potential disruption when you upgrade the adapter, you would be more likely to embed the adapter with the application. Embedding the adapter with the application allows you to associate an adapter version with an application version and administer it as a single module.

To deploy the RAR file to the application server, you must obtain and install Adapter for JD Edwards EnterpriseOne. This provides the RAR file that you install following instructions supplied with WebSphere Application Server.

Considerations for embedding an adapter in the application

Consider the following items if you plan to embed the adapter with your application:

- An embedded adapter has class loader isolation.

  A class loader affects the packaging of applications and the behavior of packaged applications deployed on run time environments. *Class loader isolation* means that the adapter cannot load classes from another application or module. Class loader isolation prevents two similarly named classes in different applications from interfering with each other.

- Each application in which the adapter is embedded must be administered separately.

Considerations for using a stand-alone adapter

Consider the following items if you plan to use a stand-alone adapter:

- Stand-alone adapters have no class loader isolation.

  Because stand-alone adapters have no class loader isolation, only one version of any given Java artifact is run and the version and sequence of that artifact is undetermined. For example, when you use a stand-alone adapter there is only *one* resource adapter version, *one* adapter foundation class (AFC) version, or *one* third-party JAR version. All adapters deployed as stand-alone adapters share a single AFC version, and all instances of a given adapter share the same code version. All adapter instances using a given third-party library must share that library.
- If you update any of these shared artifacts, all applications using the artifacts are affected.

  For instance, if you have an adapter that is working with server version X, and you update the version of the client application to version Y, your original application might stop working.
- Adapter Foundation Classes (AFC) is compatible with previous versions, but the latest AFC version must be in every RAR file that is deployed in a stand-alone manner.

  If more than one copy of any JAR file is in the class path in a stand-alone adapter, the one that is used is random; therefore, they all must be the latest version.

Considerations while deploying adapters with different versions

When you install multiple adapters with different versions of CWYBS_AdapterFoundation.jar, and if a lower version of the CWYBS_AdapterFoundation.jar is loaded during runtime, the adapter will return the ResourceAdapterInternalException error message, due to a version conflict. For example, when you install Oracle E-Business Suite adapter version 7.0.0.3 and WebSphere Adapter for JD Edwards EnterpriseOne version 7.5.0.2, the following error message is displayed "The version of CWYBS_AdapterFoundation.jar is not compatible with IBM WebSphere Adapter for JD Edwards EnterpriseOne" as IBM WebSphere Adapter for JD Edwards EnterpriseOne loads file:/C:/IBM/WebSphere/ProcServer7/profiles/ProcSrv01/installedConnectors/CWYOE_OracleEBS.rar/CWYBS_AdapterFoundation.jar with version 7.0.0.3. However, the base level of this jar required is version 7.5.0.2. To overcome this conflict, you must ensure that all adapters are at same version level. For further assistance, contact WebSphere Adapters Support for help.

There are occasions when you have to work with embedded adapters that do not need a client-server communication, standalone adapters that need a server connection, or a hybrid mix of adapter connections.

The following scenarios cover the different behaviors of AFC version conflict detection, when you are deploying two or more adapters and at least one of the adapter version is 7.5 or higher.

Deploying a standalone Adapter

1. Install WebSphere Adapter for Siebel Business Applications version 7.0.1.0 through the WebSphere Application Server administrative console.
2. Install WebSphere Adapter for SAP Software version 7.5.0.0 through the administrative console.
3. Create ActivationSpec for an ALE passthrough inbound operation.
4. Create an application in Rational Application Developer for WebSphere Software for a standalone ALE passthrough inbound operation.
5. Install and start the application through the administrative console.
6. Verify the error.

Note: An error message will be generated in the log/trace area of WebSphere Application Server, to indicate an AFC version conflict.

Deploying an embedded Adapter

1. Import a build of WebSphere Adapter for PeopleSoft Enterprise version 7.0.1.0, using a RAR file.
2. Create a Peoplesoft Inbound EMD operation.
3. Import a build of WebSphere Adapter for Oracle E-Business Suite version 7.5.0.0, using a RAR file.
4. Create an Oracle inbound EMD operation, in the same module where you have created the Peoplesoft Inbound EMD operation.
5. Deploy the module to WebSphere Application Server.
6. Check the trace.

At step 5, the deployment will fail. At step 6, you will get an internal error message due to the AFC version conflict.

Note: To avoid a name conflict between the business object generated by the two adapters, you may need to generate the artifacts into different folders.

Deploying a combination of standalone and embedded Adapters

1. Install WebSphere Adapter for Oracle E-Business Suite version 7.0.1.0 through theWebSphere Application Server administrative console.
2. Create an ActivationSpec for a Oracle inbound operation.
3. Create an application in Rational Application Developer for WebSphere Software for a Oracle inbound operation, for the standalone Adapter deployment.
4. Deploy the Oracle inbound application and trigger your inbound events.
5. Create an application in Rational Application Developer for WebSphere Software for a WebSphere Adapter for SAP Software version 7.5.0.0 inbound embedded Adapter deployment.
6. Deploy an SAP inbound application, and trigger your inbound events.

Note: You can resolve the AFC version conflict by using different class loaders for the standalone and embedded deployments. You can start both Oracle and SAP inbound applications successfully, and process Inbound events without exception.

For further assistance, visit http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.

WebSphere Adapters in clustered environments

You can improve adapter performance and availability by deploying a module on a clustered server environment. Clusters are groups of servers that are managed together to balance workloads and to provide high availability and scalability.

The module you deployed is replicated across all servers in a cluster, regardless of whether you deploy the module using a stand-alone or an embedded adapter. The following IBM products support WebSphere Adapters in a clustered environment:

- WebSphere Application Server
- WebSphere Application Server Network Deployment
- WebSphere Extended Deployment

To deploy and configure WebSphere Adapter for JD Edwards EnterpriseOne in a clustered environment, see: "Deploying the module in a clustered environment" on page 52. When you set up a server cluster, you create a Deployment Manager profile. The HAManager, a subcomponent of the Deployment Manager, notifies the Java Platform, Enterprise Edition (JEE) Connector Architecture (JCA) container to activate an adapter instance.

Using WebSphere Extended Deployment, you can optionally enhance the performance of adapter instances in your clustered environment. WebSphere Extended Deployment extends the WebSphere Application Server Network Deployment capabilities by using a dynamic Workload Manager instance instead of a static Workload Manager. The dynamic Workload Manager instance can optimize the performance of adapter instances in the cluster by dynamically balancing the load of the requests. This means that application server instances can be automatically stopped and started based on the load variations, allowing systems with different capacities and configurations to handle load variations evenly.

In clustered environments, adapter instances for WebSphere Adapter for JD Edwards EnterpriseOne can handle outbound processes only.

High availability for inbound processes

Inbound processes are based on events triggered as a result of updates to data in the JD Edwards EnterpriseOne server. WebSphere Adapter for JD Edwards EnterpriseOne is configured to detect updates by polling an event table. The adapter then publishes the event to its endpoint.

When you deploy a module to a cluster, the Java Platform, Enterprise Edition (JEE) Connector Architecture (JCA) container checks the enableHASupport resource adapter property. If the value for the enableHASupport property is true, which is the default setting, all of the adapter instances are registered with the HAManager with a policy 1 of N. This policy means that only one of the adapter instances starts polling for events. Although other adapter instances in the cluster are started, they remain dormant with respect to the active event until the active adapter instance finishes processing the event. If the server on which the polling thread was started shuts down for some reason, an adapter instance that is running on one of the backup servers is activated.

Note: In the active-passive configuration mode of the adapters, the endpoint application of the passive adapter instance also listens to the events/messages even if the `enableHASupport` property is set to `True`. This is because the `alwaysactivateAllMDBs` property in the JMS activation specification is set to `True`. To stop the endpoint application of the passive adapter instance from listening to

the events, you must set the `alwaysactivateAllMDBs` property value to `False`. For more information, see Disabling end point applications of the passive adapter .

Note: In clustered environments, when the adapter works in a Active-Active configuration, it provides both high availability and load balancing support. This functionality is useful in production environments where high performance is needed.

Important: Do not change the setting of the enableHASupport property.

High availability for outbound processes

In clustered environments, multiple adapter instances are available to perform outbound process requests. Accordingly, if your environment has multiple applications that interact with WebSphere Adapter for JD Edwards EnterpriseOne for outbound requests, then you might improve performance by deploying the module to a clustered environment. In a clustered environment, multiple outbound requests can be processed simultaneously, as long as they are not attempting to process the same record.

If multiple outbound requests are attempting to process the same record, such as a Customer address, the workload management capability in WebSphere Application Server Network Deployment distributes the requests among the available adapter instances in the sequence they were received. As a result, these types of outbound requests in a clustered environment are processed in the same manner as those in a single server environment: one adapter instance processes only one outbound request at a time.

Chapter 3. Configuring the module for deployment

To configure the adapter so that it can be deployed on WebSphere Application Server, use Rational Application Developer for WebSphere Software to create a module, which is exported as an EAR file when you deploy the adapter. You then specify the business objects you want to discover and the system on which you want to discover them.

Launching the J2C Bean wizard

To begin the process of creating and deploying a module, you start the J2C Bean wizard in Rational Application Developer for WebSphere Software. The wizard creates a connector project, which is used to organize the files associated with the module.

Before you begin

Ensure that you have gathered the information you need to establish a connection to the JD Edwards EnterpriseOne server. For example, you need the name or IP address of the JD Edwards EnterpriseOne server and the user ID and password to access it.

About this task

If you have an existing project, you can use it instead of creating a new one. Select it before you start the wizard.

Procedure

1. To start the J2C Bean wizard, go to the Enterprise Explorer of Rational Application Developer for WebSphere Software and click **File** > **New** > **Other** > **J2C** > **J2C Bean**. Click **Next**.
2. In the Resource Adapter Selection window, expand the JD Edwards EnterpriseOne folder and select **IBM WebSphere Adapter for JD Edwards EnterpriseOne (IBM : *version*)**, where *version* is the version of the adapter you want to use., and then click **Next**.
3. In the **Connector Import** window, accept the default project name in the **Connector project** field or type a different name.
4. In the **Target server** field, select the type of server where you want to deploy the module. The wizard creates the artifacts that are appropriate to that server.
5. Click **Next**. The Connector Settings window is displayed.

What to do next

Continue working in the J2C Bean wizard. The next step is to add dependent JAR files to the project.

Configuring the connector dependencies

The JD Edwards EnterpriseOne application requires that you add software dependency files to the project. These software dependencies enable the J2C Bean wizard to communicate with the JD Edwards EnterpriseOne environment.

Before you begin

Create the project, or select an existing project.

About this task

To obtain the required software dependency files and specify their location, use the following procedure.

Procedure

1. Obtain the JD Edwards EnterpriseOne software dependency files from your JD Edwards EnterpriseOne administrator. The necessary files are listed in the following table.

   Note: The software dependencies differ, depending on which version of JD Edwards EnterpriseOne Tools you use.

Table 2. Connector dependency files required by JD Edwards EnterpriseOne Tools

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
connector.jar	Common_Jar.jar	ApplicationAPIs_JAR.jar	ApplicationAPIs_JAR.jar
database.jar	Connector.jar	ApplicationLogic_JAR.jar	ApplicationLogic_JAR.jar
jdeinterop.ini	database.jar	Base_JAR.jar	Base_JAR.jar
jdeLog.properties	EventProcessor_EJB.jar	BizLogicContainer_JAR.jar	BizLogicContainerClient_JAR.jar
kernel.jar	jdeutil.jar	BizLogicContainerClient_JAR.jar	BizLogicContainer_JAR.jar
log4j.jar	jdbj.ini	bootstrap.jar	BusinessLogicServices_Jar.jar
owra.jar	jdeinterop.ini	castor.jar	castor.jar
xalan.jar	jdelog.properties	Connector.jar	commons-httpclient-3.0.jar
xerces.jar	kernel.jar	ecutils.jar	commons-logging.jar
JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	log4j.jar	EventProcessor_JAR.jar	Connector.jar
	xalan.jar	EventProcessor_EJB.jar	EventProcessor_EJB.jar
	xerces.jar	jdbj.ini	EventProcessor_JAR.jar
	JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	JdbjBase_JAR.jar	Generator_JAR.jar
		JdbjInterfaces_JAR.jar	jdbj.ini

Table 2. Connector dependency files required by JD Edwards EnterpriseOne Tools (continued)

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
		jdeinterop.ini	JdbjBase_JAR.jar
		jdelog.properties	JdbjInterfaces_JAR.jar
		JdeNet_JAR.jar	jdeinterop.ini
		lmproxy.jar	jdelog.properties
		log4j.jar	JdeNet_JAR.jar
		messagingClient.jar	jmxremote.jar
		naming.jar	jmxremote_optional.jar
		PMApi_JAR.jar	jmxri.jar
		Spec_JAR.jar	log4j.jar
		System_JAR.jar	ManagementAgent_JAR.jar
		urlprotocols.jar	Metadata.jar
		xalan.jar	MetadataInterfaces_JAR.jar
		xerces.jar	PMApi_JAR.jar
		JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	Spec_JAR.jar
			SystemInterfaces_JAR.jar
			System_JAR.jar
			xalan.jar
			xerces.jar
			xmlparserv2.jar
			JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • classes12.zip

2. Copy the software dependency files to a temporary location. For example, copy them to `C:\temp\JDE_dependencies\`.

   Tip: If you put the JDBC driver files in their own folder, it makes them easier to browse to from the J2C Bean wizard. For example, if you are using an Oracle database server, you can specify the `tnsnames.ora` and the `classes12.zip` files in the following location: `C:\temp\JDE_dependencies\jdbc_driver\`.
3. In the Required Files and Libraries window, use the following procedure to add the software dependency files to the project.

a. Select the version of JD Edwards EnterpriseOne Tools from the left pane. The required files for the version you choose are displayed in the right pane.
b. To locate and select the required JAR, INI, and PROPERTIES files, click **Browse**, navigate to the folder that contains the files listed, then click **OK**.
c. To add the JDBC driver files, click **Add**, navigate to the location of the JDBC driver files, select the `tnsnames.ora` and `classes12.zip` files, then click **OK**.

4. Click **Next**.

Results

The adapter is configured with the names of the files it needs to communicate with the JD Edwards EnterpriseOne server.

What to do next

Verify or edit the connection information in some of the software dependency files to make sure the J2C Bean wizard can connect with the JD Edwards EnterpriseOne server.

Editing connector dependency files

Some of the software dependency files for JD Edwards EnterpriseOne contain editable information, such as the Rational Application Developer for WebSphere Software workplace location and JD Edwards EnterpriseOne server name and port number. This type of information may need to be edited before you can establish a connection between the J2C Bean wizard and the JD Edwards EnterpriseOne server.

Before you begin

Make sure you have added the software dependency files to your project and that you know the location of the Rational Application Developer for WebSphere Software workplace.

About this task

Verify that the information in your software dependency files correctly reflects the Rational Application Developer for WebSphere Software workplace location and the connection information required by the JD Edwards EnterpriseOne environment. To do this, use the following procedure.

For further information about configuring the software dependency files, refer to the *JD Edwards EnterpriseOne Tools Connectors* documentation for your version of JD Edwards EnterpriseOne.

Procedure

1. If the J2C Bean wizard is open, click **Cancel** to close it.
2. Verify the information in the jdbj.ini file.
   a. Double-click **jdbj.ini** to open it. The file opens in Notepad.
   b. Press **Ctrl+F** to search the file.
   c. Enter `tns` in the Find what field.

d. Verify that the location listed for the `tnsnames.ora` file is the correct location for the Rational Application Developer for WebSphere Software workplace for this project. For example: `tns=C:\IBM\wid6.1\workspace\CWYED_JDE\connectorModule\tnsnames.ora`
e. Click **File** > **Save** to save any changes you made to the file.

3. Verify the information in the jdeinterop.ini file.
   a. Double-click **jdeinterop.ini** to open it. The file opens in Notepad.
   b. Verify that the server name and the port number are correct. You can obtain this information from the JD Edwards EnterpriseOne administrator.
   c. Click **File** > **Save** to save any changes you made to the file.
4. Verify the information in the jdelog.properties file.
   a. Double-click **jdelog.properties** to open it. The file opens in the right-pane of Rational Application Developer for WebSphere Software.
   b. Verify that the information in this file is correct. This file specifies log levels and log file locations. You can obtain this information from the JD Edwards EnterpriseOne administrator.
   c. Click **File** > **Save** to save any changes you made to the file.

Results

The wizard has the files it needs to connect to the JD Edwards EnterpriseOne server.

What to do next

In the Discovery and Configuration window, specify the connection properties that the J2C Bean wizard needs to communicate with the JD Edwards EnterpriseOne environment.

Setting connection properties for the J2C Bean wizard

To set connection properties for the J2C Bean wizard so that it can access the JD Edwards EnterpriseOne server, specify such information as the user name and password you use to access the server as well as the environment name and role name required by the JD Edwards EnterpriseOne environment.

Before you begin

Make sure you have successfully added the external software dependency files, and that you have edited the connection information in the dependency files.

About this task

Specify the connection properties that the J2C Bean wizard needs to connect to the JD Edwards EnterpriseOne environment and discover its business objects and services.

Note: You can optionally set bidirectional properties and logging properties in the same J2C Bean wizard window as you set the connection properties.

To specify the required connection properties and optional bidirectional and logging properties, use the following procedure.

Procedure

1. In the Adapter Style window, select **Outbound** to pass data from your service import to the adapter or **Inbound** to pass data from the adapter to your service export, and then click **Next**. The Discovery Configuration window is displayed.
2. Optional: Event monitoring is available to your application only if you have Business Monitor or WebSphere Business Events installed in your environment.
   - Use the following procedure, to generate and monitor common base events and manage these events with the Common Event Infrastructure (CEI) services using Business Monitor:
     a. Select the Enable Inbound Event Monitor check box and then click **Next**. The **Event and JMS configuration** window is displayed.
     b. In the **Event type** field, select WebSphere Business Monitor.
     c. In the **Queue connectionFactory JNDI name** field, accept the default value, `jms/cei/EventQueueConnectionFactory`.
     d. In the **Queue JNDI name** field, accept the default value, `jms/cei/EventQueue`.
     e. Click **Advanced** to add advanced properties.
     f. In the Remote JNDI provider configuration section, specify the naming provider URL host name and port name values in the **Naming provider URL Host** and **Naming provider URL port** fields to connect to the remote WebSphere Application Server from the wizard.
     g. In the Connection authentication configuration section, type the user name in the **User name** field to connect to the server from the wizard.
     h. In the Connection authentication configuration section, type the password in the **Password** field to connect to the server from the wizard.
     i. Click **Next**. The Discovery Configuration window is displayed.
   - Use the following procedure, to generate and monitor common base events and manage these events using WebSphere Business Events:
     a. Select the Enable Inbound Event Monitor check box and then click **Next**. The **Event and JMS configuration** window is displayed.
     b. In the **Event type** field, select WebSphere Business Events.
     c. In the **Topic connectionFactory JNDI name** field, accept the default value, `jms/WbeTopicConnectionfactory` .
     d. In the **Topic JNDI name** field, accept the default value, `jms/WBE/CbeListener`.
     e. Click **Advanced** to add advanced properties.
     f. The **Remote JNDI provider configuration** field is used to configure the remote topics.

        If the bus in the local cell has the same name as the bus in remote cell, the application always connect to the local cell. It does not use any of the provider endpoints specified on the connection factory, so the Remote Topic Configuration information that you enter is ignored. For more information about remote Topic Configuration, refer to http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.pmc.doc/tasks/tjn0033_.html .
     g. In the **Naming provider URL Host** field, specify the naming provider URL host name to connect to the remote WebSphere Application Server from the wizard.

h. In the **Naming provider URL port** field, specify the naming provider URL port name to connect to the remote WebSphere Application Server from the wizard.
i. In the Connection authentication configuration section, type the user name in the **User name** field to connect to the server from the wizard.
j. In the Connection authentication configuration section, type the password in the **Password** field to connect to the server from the wizard.
k. Click **Next**. The Discovery Configuration window is displayed.

3. In the Adapter Style window, specify the configuration properties:
   a. In the Environment field, type the name of the JD Edwards EnterpriseOne environment.
   b. In the Role field, type the role name you use to access the JD Edwards EnterpriseOne environment.
   c. In the User name field, type the user name required to access the JD Edwards EnterpriseOne server.
   d. In the Password field, type the password you use to access the JD Edwards EnterpriseOne server.
4. Optional: To set additional advanced properties, click **Advanced**.
5. Optional: To enable bidirectional support for the adapter at run time:
   a. In the **Bidi properties** area, select **Bidi transformation**.
   b. Set the ordering schema, text direction, symmetric swapping, character shaping, and numeric shaping properties to control how bidirectional transformation is performed.
6. Optional: To change the location of the log files for the wizard or the amount of information included in the logs, click **Specify the level of the logging desired**, and then provide the following information:
   a. In **Log file output location**, specify the location of the log file for the wizard.
   b. In **Logging level**, specify the severity of errors that you want logged.

      Note: This log information is for the wizard only; at run time, the adapter writes messages and trace information into the standard log and trace files for the server.
7. Click **Next**.

Results

The J2C Bean wizard contacts the JD Edwards EnterpriseOne server, using the information you provided (such as user name and password) to log in. You see the Object Discovery and Selection window.

What to do next

Specify search criteria that the J2C Bean wizard uses to discover functions or data on the JD Edwards EnterpriseOne server.

Configuring the module for outbound processing

To configure a module to use the adapter for outbound processing, use the J2C Bean wizard in Rational Application Developer for WebSphere Software to find and select business objects and services from the JD Edwards EnterpriseOne server, and to generate the business object definitions and related artifacts.

Generating business functions

To configure WebSphere Adapter for JD Edwards EnterpriseOne using business functions, use the J2C Bean wizard in Rational Application Developer for WebSphere Software to select business functions that are in the JD Edwards EnterpriseOne server, and generate business object definitions and related artifacts for outbound processing.

Selecting business objects and services

To specify which business function you want to call and which data you want to process, you provide information in the J2C Bean wizard.

Before you begin

Make sure you have set the connection properties for the J2C Bean wizard.

About this task

Specify search criteria that the J2C Bean wizard uses to discover business functions on the JD Edwards EnterpriseOne server. The J2C Bean wizard returns a list of business functions that meet the search criteria.

To specify the search criteria and select a business function, use the following procedure.

Procedure

1. In the Object Discovery and Selection window, make sure the query is set up to find business objects in the JD Edwards EnterpriseOne server. In the Query field, make sure you see **Business Functions=true**. If you see **Business Functions=false**, use the following procedure to set Business Functions to true.
   a. Click **Edit Query**.
   b. In the Query Properties window, clear the check box for XML Lists so that query finds only business functions. The **Add** button in the Query Properties window is grayed out.
   c. Click **OK**.
2. Click **Run Query**.
3. In the Discovered objects list, select business function node and expand it.
   a. Expand **Business functions**. This enables the filter button.
   b. Search for business functions using one of the following methods.
      - Click the filter button if you know the Library name (for example, **CFIN**), the C File name (for example, **B01000033**), and the business function name (for example, GetEffectiveAddress) for the business function you want.
      - If you do not know the Library name, C File name, or business function name, expand **Business functions**, expand the Library name, expand the C File name, then select the business function.

        Tip: The mouseover text indicates the purpose of each Library name and C File name.
4. Select the business function. For example, if you navigated to **Business Functions** > **CFIN** > **B0100033**, select **GetEffectiveAddress**.
5. Click the arrow button to add the business function to the **Selected objects** list.

6. In the window, either accept the default business object name or type a different name. The default name is the name of the business function.

   Note: The business object name has no semantic value, so you can give it a meaningful name, and the name you assign it will not impact how the business object functions.
7. Click **OK** to add the business function to the list of business objects to be imported.
8. Click **Next**.

Results

You have selected the business function you want to work with and selected a name for it.

What to do next

From the Configure Composite Properties window, specify a business object container name and associated operation. Optionally specify a namespace and directory to which the generated business object will be stored and indicate whether you want a business graph generated.

Configuring the selected objects

To configure the business function, you specify information about the object, such as the name of the object and the operation associated with the object.

Before you begin

Make sure you have selected and imported the business function.

About this task

To configure the business function, use the following procedure.

Procedure

1. Optional: In theConfigure Composite Properties window, enter the following information.
   a. In the **Business object namespace** field, use the default namespace (http://www.ibm.com/xmlns/prod/websphere/j2ca/jde) except in the following circumstance: if you are adding the business object to an existing module and the module already includes that business object (from an earlier run of the J2C Bean wizard), change the namespace value. For example, you could change the namespace to http://www.ibm.com/xmlns/prod/websphere/j2ca/jde1.
   b. In the **Relative path** field, specify the directory to store the business object.
2. Required: Add a business object container to the business function, and assign a name to the business object container. All business functions require that you add a business object container.

   Note: You can add one or many business functions inside a single business object container.
   a. Click the **Add** button.

b. In the **Add window**, type a name in the **Value** field. You can type any meaningful name. For example, if the business function is called GetEffectiveAddress, you can type **GetEffectiveAddressContainer**.

3. Required: In the **Operations Field**, specify the **Execute** operation to be associated with the business object.
4. Required: Associate the business functions with an operation.
   a. Click the **Add** button to create a business function.
   b. In the **Add** window, select the corresponding the business function, then click **OK**.
5. Optional: Run a business function multiple times.
   a. Select the corresponding business function.
   b. Click the **Add** button to specify the properties for the business function.
   c. Select the **Array** check box, to run the business function multiple times.
   d. Select the **RollBackOnWarning** check box, to run the business function even if the business function returns warnings.
   e. Select the **RunOnError** check box, to run the business function even if the business function returns an error.
6. Click **Next**.

Results

You specified a name for the business object container and selected an operation for the business function. The J2C Bean Creation and Deployment Configuration window is displayed.

What to do next

Generate a deployable module that includes the adapter and the business objects.

Setting deployment properties and generating artifacts

After you select and configure business objects for your module, use the J2C Bean wizard to configure properties that the adapter uses to connect to a specific database. The wizard creates a project where all the artifacts and property values are saved.

About this task

This the common content that can be added for all RAD adapters.

Procedure

1. In the **Project name** field, select or create a project into which the J2C Bean is generated.
   - To select an existing project, click **Browse**. If the project name you want appears in the **Project Selection** list, select its name.
   - Otherwise, click **New** to create a project. In the New Source Project Creation window, select the type of project you want to create.
     a. Java project: For information about creating a Java project, see Creating a Java project.
     b. EJB project: For information about creating a EJB project, see Creating an EJB project.
     c. Web project: For information about creating a web project, see Creating a web project.

2. In the **Package name** field, select or create a package into which the J2C Bean is generated.
   - To select an existing package, click **Browse**. If the package name you want appears in the **Project Selection** list, select its name.
   - Otherwise, create a package:
     a. Click **New**.
     b. In the Java Package window, type a name for the package. For example, myadapteroutboundpkg.
     c. Click **Finish**.
3. In the **Interface name** field, specify the interface name you want to use for your business objects. For example, MyAdapterOutboundInterface. The implementation name is automatically generated by suffixing "Impl" to the interface name, and the name is displayed in the **Implementation name** field. For example, MyAdapterOutboundInterfaceImpl.
4. Optional: In the **Generate Command Bean** section, select the operation for which you want to generate a command bean. If you create command bean, you need to specify the command bean name as well as the input and output names.
5. In the Connection properties area, specify how you want the adapter to connect to the database.
   - In the managed connection mode, you can specify connection properties in the Java Naming and Directory Interface (JNDI) name of the managed connection factory defined on the application server. The managed connection factory is EIS adapter-specific and contains all of the required connection information. Managed connection implies that the resource adapter is installed directly on the server and, therefore, the JNDI name of the managed connection factory is visible to all EAR (enterprise application archive) files installed on the application server. To obtain the connection through JNDI, select the **Managed Connection (recommended)** check box. This type of connection is managed by the application server.
   - In the non-managed connection mode, the application communicates directly to the EIS and manages all the connections. Therefore, all of the connection information in this setup is encapsulated within the application. For a J2C bean, this means that all of the connection information is specified in the generated J2C bean. To obtain the connection directly from the resource adapter, select the **Non-managed Connection** check box.

   Note: If you check both choices, the Managed Connection is executed first. If the JNDI name cannot be found, the non-managed connection parameters is used.
6. If you select **Managed Connection (recommended)** check box, you specify the connection properties in the JNDI name of the managed connection factory defined on the application server. You can either select an existing JNDI or create a new one.
   - To select an existing JNDI name:
     a. Click **Browse**.
     b. In the Server selection window, specify the server on which to deploy the adapter and click **Next**.
     c. Select the existing JNDI name in the J2C Connection Factories window. The existing RAR and the JNDI name are used for the connection factory.
   - To create a new one, click **New**.

a. In the Server selection window, specify the server on which to deploy the adapter and click **Next**.
b. In the New J2C Connection Factory window, specify the name in the **JNDI Name** field.
c. Specify the advanced properties by clicking **Advanced**. The RAR file is installed on IBM WebSphere Application Server and the connection factory is created using the JNDI name specified.
d. Expand each of the following group sections to review the advanced properties:
   – **Logging and tracing**
     - If you have multiple instances of the adapter, set **Adapter ID** to a value that is unique for this instance.
     - If you want to mask sensitive information in log and trace files (for example, if you want to avoid making customer information visible in these files), select **Disguise user data as "XXX" in log and trace files**.
e. Click **Finish** to return to the J2C Bean Creation and Deployment Configuration window.

7. If you select the **Non-managed Connection** check box, the system connection information area is expanded to show the connection information. Review the connection information and change the values if required. Specify the advanced properties by clicking **Advanced**. Refer to the advanced properties defined for managed connection.
8. When you are finished setting properties, select the **Click to launch the J2C deployment** checkbox to immediately start the wizard.
9. Click **Finish**.

Results

Generating the EJB or JSP project

After you create the J2C bean, use the Web Page, Web Service, or EJB from J2C Java wizard to create the JSP, EJB or web service.

About this task

You can create a JSP, an EJB, or a web service to deploy your J2C bean.

Procedure

1. To start the J2C Bean wizard, go to the Enterprise Explorer of Rational Application Developer for WebSphere Software and click **New>Other>J2C> Web Page, Web Service, or EJB from J2C Java Bean**.
2. Click **Next**.
3. In the J2C Java bean selection window, click **Browse**.
4. In the Find J2C bean window, type the first letter of the implementation name generated earlier or type the full name and press the Enter key.
5. Select the implementation name from the list and click **OK**.
6. Click **Next**.
7. In the Deployment Information window, select the Java EE Resource Type as **EJB**, **Simple JSP** or **Web Service** and click **Next**.

Note: In the Deployment Information window, the **Configure Resource Adapter Deployment** check box is available for selection only if you have selected the **Non-managed Connection** check box when specifying the deployment settings.

8. If you select **EJB**, the Enterprise bean creation wizard is displayed. This wizard creates the Java project as an EJB. For information about creating an EJB, see Deploying to an EJB.

   Note: You can deploy an EJB only when the J2C bean is in the EJB project. Otherwise, this option is not available for selection.
9. If you select **Simple JSP**, the Simple JSP Creation wizard is displayed. For details about creating a new web project, see Creating a web project. For information about creating a Simple JSP, see Deploying to a Simple JSP.
10. If you select **Web Service**, the Web Service Creation wizard is displayed. For information about creating a web service, Deploying a J2C application as a web service.
11. Export the project as an EAR file for deployment.

Generating XML Lists

To configure WebSphere Adapter for JD Edwards EnterpriseOne using XML Lists, use the J2C Bean wizard in Rational Application Developer for WebSphere Software to select XML Lists that are in the JD Edwards EnterpriseOne server, and generate business object definitions and related artifacts for outbound processing.

Selecting business objects and services

To specify which data you want to process for the XML List, you provide information in the J2C Bean wizard.

Before you begin

Make sure you have set the connection properties for the J2C Bean wizard. Also, make sure you have gathered the table information, including table names and table types, for the XML Lists you want to create.

Tip: Use the JD Edwards EnterpriseOne Universal Table Browser (UTB) to gather the table information for the XML Lists you want to create. For information about using the JD Edwards EnterpriseOne UTB, refer to the *JD Edwards EnterpriseOne Tools 8.96 System Administration Guide*.

About this task

Specify search criteria that the J2C Bean wizard uses to discover database table information on the JD Edwards EnterpriseOne server. The J2C Bean wizard returns a list of objects that meet the search criteria.

To specify the search criteria for creating an XML List, use the following procedure.

Procedure

1. In the Configure Composite Properties window, edit the query to prepare it to find the table information that is required for creating the XML List.
   a. Click **Edit Query**.
   b. In the Query Properties window, select **XML Lists**.

Note: You can optionally clear the check box for **Business Functions** so the query finds only XML Lists. If you leave **Business Functions** selected, the query returns both XML Lists and business functions.

c. To add the table name, click **Add**.

d. In the **Add** window, enter the name of the database table you want to add to the query. For example, enter `F0116` for the database table. If you do not know the name of the table you need, you can use the JD Edwards EnterpriseOne Universal Table Browser (UTB) to find it on the JD Edwards EnterpriseOne server.

e. Click **OK** in the Add window. The table name you entered is displayed in the Tables list.

f. Click **OK** in the Query Properties window.

The query is now ready to discover the JD Edwards EnterpriseOne database table you specified in order to create the XML List.

2. Run the XML List query to find and discover the table in the JD Edwards EnterpriseOne server that matches the table you specified.

   a. Click **Run Query**. The J2C Bean wizard queries the JD Edwards EnterpriseOne server. The results of the query appear in the Discovered objects list.

   b. In the Discovered objects list, expand the **XML Lists**.

   c. Navigate to the table that matches the table you created.

   d. Click the arrow button to add the table to the **Selected objects** list.

3. In the Configure Composite Properties window, add search criteria to the XML List query before importing the data from the JD Edwards EnterpriseOne server. This allows you to specify query parameters such as table type and sorting conditions.

   a. In the Business Object Name field, you can either keep the default business object name or rename it to suit your needs. The default name is the name of the table.

   Note: The business object name has no semantic value, so you can give it a meaningful name, and the name you assign it will not impact how the business object functions.

   b. In the Table type field, select the type of table from the list. For example, select OWTABLE table type for the F0116 table.

   The following table types are available.

Table 3. Table types for XML Lists

Table type	Description	When to use
OWTABLE	A JD Edwards EnterpriseOne database table	Use this table type if the table you want is located in the JD Edwards EnterpriseOne database.
OWVIEW	A business view used to define the relationship between two or more tables and joins the data into a single view	Use this business view as input to the TABLE_CONVERSION table type.
FOREIGN_TABLE	A non-JD Edwards EnterpriseOne database table that resides in a database supported by JD Edwards EnterpriseOne, such as Oracle, Access, iSeries®, or SQL Server	Use this table type if the table you want is located in a non-JD Edwards EnterpriseOne database.

Table 3. Table types for XML Lists (continued)

Table type	Description	When to use
TABLE_CONVERSION	A table type that uses batch processes which allow you to rapidly manipulate data in tables. You can set up table conversions as templates, running them multiples times, then revising them using different versions to suit the needs of your environment.	Use this table type if you plan to manipulate batches of data in one of the following ways: • Data Conversion: allows you to transfer data from an input table or business view into output tables; also allows you to update records in a table or business view • Data Copy: allows you to copy tables from one data source or environment to another data source or environment when the tables are identical • Data Copy with Table Input: allows you to copy tables based on information from an input table. • Batch Delete: allows you to delete records from a table or business view

c. To add a sorting condition to the query, click **Add Sorting**, click **Select** to select the attribute you want sorted, then select **ASCENDING** or **DESCENDING** from the Sorting list.

   Note: If you want to remove a sorting condition, click **Remove Sorting Condition**.

d. To add other conditions to the query, click **Add Condition**, then select one of the following conditions.
   - **Attribute**: Select the attribute for which you want to add a condition.
   - **Clause**: Select the clause for the query condition. The default is **Where**.
   - **Operator**: Select the operator when comparing the attribute to the column value.
   - **Use Attribute Value**: Select an attribute to compare to.
   - **Default**: Specify the default value for the query condition.

   Note: If no conditions are specified, all records are retrieved. If you want to add multiple conditions to the query, click **Add Condition** again. If you want to remove any unwanted conditions, click **Remove condition**.

e. Click **OK**. The table name is displayed in the Selected objects list.

4. Click **Next**.

Results

You have selected the table you want to work with and configured it for the XML List you want to create.

What to do next

From the Configure Composite Properties window, you can optionally specify a namespace and directory to which the generated XML List will be stored.

Configuring the selected objects

To configure the XML List, you need to specify information about the object, such as the maximum number of records to retrieve.

Before you begin

Make sure you have selected and imported the XML List.

About this task

To configure the XML List, use the following procedure.

Procedure

1. Optional: In the Configure Composite Properties window.
   a. In the **Business object namespace** field, use the default namespace (http://www.ibm.com/xmlns/prod/websphere/j2ca/jde) except in the following circumstance: if you are adding the XML List to an existing module and the module already includes that XML List (from an earlier run of the J2C Bean wizard), change the namespace value. For example, you could change the namespace to http://www.ibm.com/xmlns/prod/websphere/j2ca/jde1.
   b. In the **Relative path** field, specify the directory to store the XML List.
   c. In the **Maximum number of records** field, specify the maximum number of records to retrieve when processing a **RetrieveAll** operation. The default value is 100.
   d. In the **Timeout (milliseconds)** field, specify a timeout value in milliseconds. If no value is set, the default value is 30,000 milliseconds (or 30 seconds).

   Important: The business object container name and associated operation for the XML List is set by default as follows:
   - Business object container name: *<XML_List_object>*Container
   - Operation: RetrieveAll
2. Click **Next**.

Results

You have set optional configuration parameters for the XML List object. The J2C Bean wizard automatically assigned the business object container name and associated operation for the XML List. The Generate Artifacts window opens.

What to do next

Generate a deployable module that includes the adapter and the business object.

Setting deployment properties and generating artifacts

After you select and configure business objects for your module, use the J2C Bean wizard to configure properties that the adapter uses to connect to a specific database. The wizard creates a project where all the artifacts and property values are saved.

About this task

This the common content that can be added for all RAD adapters.

Procedure

1. In the **Project name** field, select or create a project into which the J2C Bean is generated.
   - To select an existing project, click **Browse**. If the project name you want appears in the **Project Selection** list, select its name.
   - Otherwise, click **New** to create a project. In the New Source Project Creation window, select the type of project you want to create.
     a. Java project: For information about creating a Java project, see Creating a Java project.
     b. EJB project: For information about creating a EJB project, see Creating an EJB project.
     c. Web project: For information about creating a web project, see Creating a web project.
2. In the **Package name** field, select or create a package into which the J2C Bean is generated.
   - To select an existing package, click **Browse**. If the package name you want appears in the **Project Selection** list, select its name.
   - Otherwise, create a package:
     a. Click **New**.
     b. In the Java Package window, type a name for the package. For example, myadapteroutboundpkg.
     c. Click **Finish**.
3. In the **Interface name** field, specify the interface name you want to use for your business objects. For example, MyAdapterOutboundInterface. The implementation name is automatically generated by suffixing "Impl" to the interface name, and the name is displayed in the **Implementation name** field. For example, MyAdapterOutboundInterfaceImpl.
4. Optional: In the **Generate Command Bean** section, select the operation for which you want to generate a command bean. If you create command bean, you need to specify the command bean name as well as the input and output names.
5. In the Connection properties area, specify how you want the adapter to connect to the database.
   - In the managed connection mode, you can specify connection properties in the Java Naming and Directory Interface (JNDI) name of the managed connection factory defined on the application server. The managed connection factory is EIS adapter-specific and contains all of the required connection information. Managed connection implies that the resource adapter is installed directly on the server and, therefore, the JNDI name of the managed connection factory is visible to all EAR (enterprise application archive) files installed on the application server. To obtain the connection through JNDI, select the **Managed Connection (recommended)** check box. This type of connection is managed by the application server.
   - In the non-managed connection mode, the application communicates directly to the EIS and manages all the connections. Therefore, all of the connection information in this setup is encapsulated within the application. For a J2C bean, this means that all of the connection information is specified in the generated J2C bean. To obtain the connection directly from the resource adapter, select the **Non-managed Connection** check box.

Note: If you check both choices, the Managed Connection is executed first. If the JNDI name cannot be found, the non-managed connection parameters is used.

6. If you select **Managed Connection (recommended)** check box, you specify the connection properties in the JNDI name of the managed connection factory defined on the application server. You can either select an existing JNDI or create a new one.
   - To select an existing JNDI name:
     a. Click **Browse**.
     b. In the Server selection window, specify the server on which to deploy the adapter and click **Next**.
     c. Select the existing JNDI name in the J2C Connection Factories window. The existing RAR and the JNDI name are used for the connection factory.
   - To create a new one, click **New**.
     a. In the Server selection window, specify the server on which to deploy the adapter and click **Next**.
     b. In the New J2C Connection Factory window, specify the name in the **JNDI Name** field.
     c. Specify the advanced properties by clicking **Advanced**. The RAR file is installed on IBM WebSphere Application Server and the connection factory is created using the JNDI name specified.
     d. Expand each of the following group sections to review the advanced properties:
        – **Logging and tracing**
          - If you have multiple instances of the adapter, set **Adapter ID** to a value that is unique for this instance.
          - If you want to mask sensitive information in log and trace files (for example, if you want to avoid making customer information visible in these files), select **Disguise user data as "XXX" in log and trace files**.
     e. Click **Finish** to return to the J2C Bean Creation and Deployment Configuration window.
7. If you select the **Non-managed Connection** check box, the system connection information area is expanded to show the connection information. Review the connection information and change the values if required. Specify the advanced properties by clicking **Advanced**. Refer to the advanced properties defined for managed connection.
8. When you are finished setting properties, select the **Click to launch the J2C deployment** checkbox to immediately start the wizard.
9. Click **Finish**.

Results

Generating the EJB or JSP project

After you create the J2C bean, use the Web Page, Web Service, or EJB from J2C Java wizard to create the JSP, EJB or web service.

About this task

You can create a JSP, an EJB, or a web service to deploy your J2C bean.

Procedure

1. To start the J2C Bean wizard, go to the Enterprise Explorer of Rational Application Developer for WebSphere Software and click **New>Other>J2C> Web Page, Web Service, or EJB from J2C Java Bean**.
2. Click **Next**.
3. In the J2C Java bean selection window, click **Browse**.
4. In the Find J2C bean window, type the first letter of the implementation name generated earlier or type the full name and press the Enter key.
5. Select the implementation name from the list and click **OK**.
6. Click **Next**.
7. In the Deployment Information window, select the Java EE Resource Type as **EJB**, **Simple JSP** or **Web Service** and click **Next**.

   Note: In the Deployment Information window, the **Configure Resource Adapter Deployment** check box is available for selection only if you have selected the **Non-managed Connection** check box when specifying the deployment settings.
8. If you select **EJB**, the Enterprise bean creation wizard is displayed. This wizard creates the Java project as an EJB. For information about creating an EJB, see Deploying to an EJB.

   Note: You can deploy an EJB only when the J2C bean is in the EJB project. Otherwise, this option is not available for selection.
9. If you select **Simple JSP**, the Simple JSP Creation wizard is displayed. For details about creating a new web project, see Creating a web project. For information about creating a Simple JSP, see Deploying to a Simple JSP.
10. If you select **Web Service**, the Web Service Creation wizard is displayed. For information about creating a web service, Deploying a J2C application as a web service.
11. Export the project as an EAR file for deployment.

Configuring the module for inbound processing

To configure a module to use the adapter for inbound processing, use the J2C Bean wizard in Rational Application Developer for WebSphere Software to find and select business objects and services from the JD Edwards EnterpriseOne server, and to generate business object definitions and related artifacts.

Selecting business objects and services

To specify which real-time events you want to work with, you provide information in the J2C Bean wizard.

Before you begin

Make sure you have set the connection properties correctly for the J2C Bean wizard.

About this task

Specify search criteria that the J2C Bean wizard uses to discover real-time events on the JD Edwards EnterpriseOne server. The J2C Bean wizard returns a list of business functions that meet the search criteria.

To specify the search criteria and select a real-time event, use the following procedure.

Procedure

1. In the Object Discovery and Selection window, make sure the query is set up to find real-time events in the JD Edwards EnterpriseOne server. In the Query field, make sure you see the following query criteria: **Real-Time Events=true**. If you see **Real-Time=false**, use the following procedure to set it to true.
   a. Click **Edit Query**.
   b. In the Query Properties window, select **Real-Time Events**.
   c. Click **OK**.
2. Click **Run Query**.
3. In the Discovered objects list, expand the following root node: **Real-Time Events** and select one or more real-time events you want to work with.
4. Click the arrow button to add the real-time event to the **Selected objects** list.
5. In the Configuration Parameters window, you can associate an operation with the action needed for the selected real-time event. This is required for real-time events which do not already have an operation field to store the action for the real-time event. If the real-time event already has an operation field, then you need only specify the name of the operation, not the field containing the associated action.
   - To associate an operation field with the selected real-time event, use the following procedure.
     a. In the Selection field, select **Configuration with Operation Field**. The Configuration Parameters window expands.
     b. In the expanded window, enter information in the following fields:
        – List Children: All single real-time events which are contained by the container real-time event or the single real-time event itself
        – Operation Field: The operation field stores the action information associated with the operation.
          1) Click **Select** to select a value for the operation field.
          2) In the Select window, select the action for the operation, then click **OK**.
        – User-defined: Create: Leave the default number ("1") to use for mapping the Create operation to the selected action, or specify a new number.
        – User-defined: Update: Leave the default number ("2") to use for mapping the Update operation to the selected action, or specify a new number.
        – User-defined: Delete: Leave the default number ("3") to use for mapping the Delete operation to the selected action, or specify a new number.
     c. Click OK.
   - To select the real-time event without associating an operation field, use the following procedure.
     a. In the Selection field, select **Configuration without Operation Field**.
     b. In the Object Discovery and Selection window, specify the operation to use with the selected real-time event.
     c. Click **OK**.

   The selected real-time event is added to the Selected objects list.

6. Click **Next**.

Results

You have selected the real-time event you want to work with and configured the appropriate operation mapping relationship for it.

What to do next

From the Configure Composite Properties window, you specify the directory where to store the real-time events.

Configuring the selected objects

To configure the real-time event, you specify information about the object.

Before you begin

Make sure you have selected and imported the real-time event.

About this task

To configure the real-time event, use the following procedure.

Procedure

1. In the Configure Composite Properties window, specify the directory to store the business object in the **Relative path** field.
2. Click **Next**.

Results

You specified a location to store the real-time event. The J2C Bean Creation and Deployment Configuration window is displayed.

What to do next

Generate a deployable module that includes the adapter and the real-time event.

Setting deployment properties and generating artifacts

After you select and configure business objects for your module, use the J2C Bean wizard to configure properties that the adapter uses to connect to a specific database. The wizard creates a project where all the artifacts and property values are saved.

About this task

This task is performed through the J2C Bean Creation and Deployment Configuration window of the J2C Bean wizard.

Procedure

1. In the **EJB Project name** field, select or create a new EJB project.
   - To select an existing project, click **Browse**. If the required project name appears in the **Project Selection** list, select its name.
   - Otherwise, create a new project:

a. Click **New**.
b. In the EJB Project window, type a project name. For example, MyAdapterInboundEJB.
c. In the EAR Membership area, click **New** to create a new ear project.
d. In EAR Application Project window, type an EAR project name. For example, MyAdapterInboundEJBEAR
e. Click **Finish** to return to the EJB Project window.
f. Click **Finish**.

2. In the **Package name** field, select or create a new package into which the J2C Bean is generated.
   - To select an existing package, click **Browse**. If the required package name appears in the **Project Selection** list, select its name.
   - Otherwise, create a new package:
     a. Click **New**.
     b. In the New Java Package window, type a name for the package. For example, myadapterinboundejbpkg.
     c. Click **Finish**.
3. In the **Stateless Session EJB's local business interface name** field, specify the interface name you want to use for your business object. For example, MyAdapterInboundInterface. The interface name is suffixed with "MDB" and it is displayed automatically in the **Message Driven EJB name** field. For example, MyAdapterInboundInterfaceMDB. Similarly, the interface name is suffixed with "SB" and it is displayed automatically in the **Stateless Session EJB name** field. For example, MyAdapterInboundInterfaceSB.
4. In the Inbound Connection configuration area, select the Java Naming and Directory Interface (JNDI) name for an existing activation specification in WebSphere Application Server or create an activation specification. For more information about creating an activation specification in WebSphere Application Server, see "Setting activation specification properties for stand-alone adapters" on page 65.
   - To select an existing activation specification,
     a. Click **Browse**.
     b. In the Server selection window, select the server on which the resource adapter will be deployed and click **Next**.
     c. Select the existing JNDI name in the J2C Activation Specification window. The existing RAR and the JNDI name are used for the activation specification.

   Note: When you try to look up an existing JNDI in the JNDI Lookup Wizard screen, the wizard looks up only the JNDI that is created using the first RAR file. If you have more than one RAR file for the same adapter in the server, you can view only the JNDI generated using the first RAR file. The JNDI generated from the other RAR files are not looked up by the wizard.
   - To create a new activation specification, click **New**.
     a. In the Server selection window, select the server on which the resource adapter will be deployed and click **Next**.
     b. In the New J2C Activation Specification window, type a JNDI name for the activation specification.
     c. Specify the advanced properties by clicking **Advanced**. The RAR file is installed on IBM WebSphere Application Server and the activation specification is created using the JNDI name specified.

d. Expand each of the following group sections to review the advanced properties:
   - **Event polling configuration**
     1) In **Interval between polling periods**, type the number of milliseconds that the adapter waits between polling periods. For more information, see “Interval between polling periods (PollPeriod)” on page 107.
     2) In **Maximum events in polling period**, type the number of events to deliver in each polling period. For more information, see “Maximum events in polling period (PollQuantity)” on page 108.
     3) In **Retry interval if connection fails**, type the number of milliseconds to wait before trying to connect after a connection failure during polling. For more information, see “Time between retries in case of system connection failure (RetryInterval)” on page 110.
     4) In **Number of times to retry the system connection**, type the number of times to retry the connection before reporting a polling error. For more information, see “Maximum number of retries in case of system connection failure (RetryLimit)” on page 109.
     5) If you want the adapter to stop if polling errors occur, select **Stop the adapter when an error is encountered while polling**. If you do not select this option, the adapter logs an exception but continues to run. For more information, see “Stop the adapter when an error is encountered while polling (StopPollingOnError)” on page 111.
     6) You can select **Retry EIS connection on startup** if you want the adapter to retry an inbound connection that was not made to the JDE application when starting. Only communication failures to the JDE application are considered. For more information, see “Retry EIS connection on startup (RetryConnectionOnStartup)” on page 109.
   - **Event delivery configuration**
     1) In **Type of delivery**, select the delivery method. The methods are described in “Delivery type (DeliveryType)” on page 105.
     2) In **Retry limit for failed events**, specify number of times the adapter tries to reestablish an inbound connection after an error. See“Maximum number of retries in case of system connection failure (RetryLimit)” on page 109, for more information.
     3) If you want to ensure that events are delivered only once and to only one export, select **Assuredoncedelivery**. This option might reduce performance but does not result in duplicate or missing event delivery. For more information, see “Ensure once-only event delivery (AssuredOnceDelivery)” on page 106.
     4) Under **Number of connections for event delivery**, specify the minimum and maximum number of connections to use to deliver events. For more information, see “Minimum connections (MinimumConnections)” on page 108 and “Maximum connections (MaximumConnections)” on page 107.
     5) Specify the **Event Persistence Data Source (JNDI) Name**.
     6) For the adapter to persist the event state in an event persistence table located on the data source select the **Create even persistence table**. Specify the event persistence table details in **The table with**

event persistence information and **The schema name of the database where the table is stored**

a) Specify **The user name used to connect to the database**.

b) Specify **The password used to connect to the database**.

e. Click **Finish** to return to the J2C Bean Creation and Deployment Configuration window.

5. Type the existing Java Authentication and Authorization Services alias. The alias is used to retrieve the user name and password set on the configured J2C activation specification. The name is case sensitive and includes the node name.
6. In the **Service Operations** section, click **Edit Operations** to review the names of operations or add a description about the operations to be generated in the interface file.
7. Click **Finish**.

What to do next

Deploy the module and test.

Chapter 4. Deploying the module

Deploy a module to place the files that make up your module and adapter into an operational environment for testing. In Rational Application Developer for WebSphere Software, the integrated test environment features runtime support for WebSphere Application Server, depending on the test environment profiles that you selected during installation.

Deployment environments

There are test and production environments into which you can deploy modules and adapters.

In Rational Application Developer for WebSphere Software, you can deploy your modules to one or more servers in the test environment. This is typically the most common practice for running and testing projects. However, you can also export modules for server deployment on WebSphere Application Server as EAR files using the administrative console or command-line tools.

Deploying the module for testing

In Rational Application Developer for WebSphere Software, you can deploy a module that includes an embedded adapter to the test environment and work with server tools that enable you to perform such tasks as editing server configurations, starting and stopping servers and testing the module code for errors. The testing is generally performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented.

Configuring the connector dependencies

Dependent JARs have to be added to the libraries directory or packaged in the EAR.

About this task

The JARs are set in the class path and these dependent libraries have to be made available for run time when the module is deployed. There are two ways to make the dependent libraries available, one for either stand-alone deployment or embedded deployment and the other for embedded deployment only.

Configuring the connector dependencies on the server

You must copy the required software dependency files to your runtime environment before you can run your adapter applications.

About this task

To copy the required files to IBM Business Process Manager or WebSphere Enterprise Service Bus, use the following procedure.

Procedure

1. Obtain the external software dependency files from your JD Edwards EnterpriseOne administrator. The files are listed in Table 4 on page 44.

Table 4. Connector dependency files required by JD Edwards EnterpriseOne Tools

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
connector.jar	Common_Jar.jar	ApplicationAPIs_JAR.jar	ApplicationAPIs_JAR.jar
database.jar	Connector.jar	ApplicationLogic_JAR.jar	ApplicationLogic_JAR.jar
jdeinterop.ini	database.jar	Base_JAR.jar	Base_JAR.jar
jdeLog.properties	EventProcessor_EJB.jar	BizLogicContainer_JAR.jar	BizLogicContainerClient_JAR.jar
kernel.jar	jdeutil.jar	BizLogicContainerClient_JAR.jar	BizLogicContainer_JAR.jar
log4j.jar	jdbj.ini	bootstrap.jar	BusinessLogicServices_Jar.jar
owra.jar	jdeinterop.ini	castor.jar	castor.jar
xalan.jar	jdelog.properties	Connector.jar	commons-httpclient-3.0.jar
xerces.jar	kernel.jar	ecutils.jar	commons-logging.jar
JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	log4j.jar	EventProcessor_JAR.jar	Connector.jar
	xalan.jar	EventProcessor_EJB.jar	EventProcessor_EJB.jar
	xerces.jar	jdbj.ini	EventProcessor_JAR.jar
	JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	JdbjBase_JAR.jar	Generator_JAR.jar
		JdbjInterfaces_JAR.jar	jdbj.ini
		jdeinterop.ini	JdbjBase_JAR.jar
		jdelog.properties	JdbjInterfaces_JAR.jar
		JdeNet_JAR.jar	jdeinterop.ini
		lmproxy.jar	jdelog.properties
		log4j.jar	JdeNet_JAR.jar
		messagingClient.jar	jmxremote.jar
		naming.jar	jmxremote_optional.jar
		PMApi_JAR.jar	jmxri.jar
		Spec_JAR.jar	log4j.jar
		System_JAR.jar	ManagementAgent_JAR.jar
		urlprotocols.jar	Metadata.jar
		xalan.jar	MetadataInterfaces_JAR.jar

Table 4. Connector dependency files required by JD Edwards EnterpriseOne Tools (continued)

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
		xerces.jar	PMApi_JAR.jar
		JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	Spec_JAR.jar
			SystemInterfaces_JAR.jar
			System_JAR.jar
			xalan.jar
			xerces.jar
			xmlparserv2.jar
			JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • classes12.zip

2. Copy the files to the server.
   - In a testing environment in Rational Application Developer for WebSphere Software, copy the files to the ${WAS_INSTALL_ROOT}/runtimes/bi_v7/lib/ext directory.
   - In a production environment, copy the files to the ${WAS_INSTALL_ROOT}/lib/ext directory of WebSphere Application Server.

Results

The external software dependency files for JD Edwards EnterpriseOne are now part of your runtime environment.

Configuring the connector dependencies when the adapter is bundled

You must copy the dependent JAR files to the EAR application before you can run your adapter applications. You must use this method only for embedded deployment.

About this task

To obtain the required files and copy them to the EAR application, use the following procedure:

Procedure

1. From the appropriate module, go to the workspace and copy the JAR files to the directory. For example if the name of the module is ModuleName, then go to the workspace and copy the JAR files to the `ModuleNameApp/EarContent` directory.
2. Modify the adapter RAR's manifest file, `manifest.mf`, with the list of JAR files required by the adapter. Add the JAR files in the following format: Class-Path: dependantjar1.jar, dependantjar2.jar.
3. Copy the native libraries to the run time bin directory and deploy the application.

Results

The vendor software libraries are now a part of your run time environment.

Adding the module to the server

In Rational Application Developer for WebSphere Software, you can add modules to one or more servers in the test environment.

About this task

In order to test your module and its use of the adapter, you need to add the module to the server.

Procedure

1. Add the module to the server
   a. Switch to the servers view. In Rational Application Developer for WebSphere Software, select **Windows** > **Show View** > **Servers**
   a. Start the server. In the Servers tab in the lower-right pane of the Rational Application Developer for WebSphere Softwarescreen, right-click on the server, and then select **Start**.
2. When the server status is *Started*, right-click on the server, and select **Add and remove projects**.
3. In the Add and Remove Projects screen, select your project and click **Add**. The project moves from the **Available projects** list to the **Configured projects** list.
4. Click **Finish**. This deploys the module on the server.

   The Console tab in the lower-right pane displays a log while the module is being added to the server.

What to do next

Test the functionality of your module and the adapter.

Testing the module for outbound processing using the test client

Test the assembled module and adapter for outbound processing using the Rational Application Developer for WebSphere Software integration test client.

Before you begin

You need to add the module to the server first.

About this task

Testing a module is performed on the interface operations of your components, which enables you to determine whether the components are correctly implemented.

Procedure

1. Select the module you want to test, right-click on it, and select **Test** > **Test Module**.
2. For information about testing a module using the test client, see the *Testing modules and components* topic in the Rational Application Developer for WebSphere Software information center.

What to do next

If you are satisfied with the results of testing your module and adapter, you can deploy the module and adapter to the production environment.

Deploying the module for production

Deploying a module created with the J2C Bean wizard to WebSphere Application Server in a production environment is a two-step process. First, you export the module in Rational Application Developer for WebSphere Software as an enterprise archive (EAR) file. Second, you deploy the EAR file using the WebSphere Application Server administrative console.

Configuring the connector dependencies on the server

You must copy the required software dependency files to your runtime environment before you can run your adapter applications.

About this task

To copy the required files to IBM Business Process Manager or WebSphere Enterprise Service Bus, use the following procedure.

Procedure

1. Obtain the external software dependency files from your JD Edwards EnterpriseOne administrator. The files are listed in Table 5.

Table 5. Connector dependency files required by JD Edwards EnterpriseOne Tools

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
connector.jar	Common_Jar.jar	ApplicationAPIs_JAR.jar	ApplicationAPIs_JAR.jar
database.jar	Connector.jar	ApplicationLogic_JAR.jar	ApplicationLogic_JAR.jar
jdeinterop.ini	database.jar	Base_JAR.jar	Base_JAR.jar
jdeLog.properties	EventProcessor_EJB.jar	BizLogicContainer_JAR.jar	BizLogicContainerClient_JAR.jar
kernel.jar	jdeutil.jar	BizLogicContainerClient_JAR.jar	BizLogicContainer_JAR.jar
log4j.jar	jdbj.ini	bootstrap.jar	BusinessLogicServices_Jar.jar
owra.jar	jdeinterop.ini	castor.jar	castor.jar
xalan.jar	jdelog.properties	Connector.jar	commons-httpclient-3.0.jar

Table 5. Connector dependency files required by JD Edwards EnterpriseOne Tools (continued)

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
xerces.jar	kernel.jar	ecutils.jar	commons-logging.jar
JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	log4j.jar	EventProcessor_JAR.jar	Connector.jar
	xalan.jar	EventProcessor_EJB.jar	EventProcessor_EJB.jar
	xerces.jar	jdbj.ini	EventProcessor_JAR.jar
	JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	JdbjBase_JAR.jar	Generator_JAR.jar
		JdbjInterfaces_JAR.jar	jdbj.ini
		jdeinterop.ini	JdbjBase_JAR.jar
		jdelog.properties	JdbjInterfaces_JAR.jar
		JdeNet_JAR.jar	jdeinterop.ini
		lmproxy.jar	jdelog.properties
		log4j.jar	JdeNet_JAR.jar
		messagingClient.jar	jmxremote.jar
		naming.jar	jmxremote_optional.jar
		PMApi_JAR.jar	jmxri.jar
		Spec_JAR.jar	log4j.jar
		System_JAR.jar	ManagementAgent_JAR.jar
		urlprotocols.jar	Metadata.jar
		xalan.jar	MetadataInterfaces_JAR.jar
		xerces.jar	PMApi_JAR.jar
		JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • ojdbc6.jar	Spec_JAR.jar
			SystemInterfaces_JAR.jar
			System_JAR.jar

Table 5. Connector dependency files required by JD Edwards EnterpriseOne Tools (continued)

JD Edwards EnterpriseOne Tools, version 8.9 (SP1, SP2), 8.93	JD Edwards EnterpriseOne Tools, version 8.94	JD Edwards EnterpriseOne Tools, version 8.95, 8.96	JD Edwards EnterpriseOne Tools, version 8.97, 8.98
			xalan.jar
			xerces.jar
			xmlparserv2.jar
			JDBC driver files For example, if you are using an Oracle database server, use the following JDBC driver files: • tnsnames.ora • classes12.zip

2. Copy the files to the server.
   - In a testing environment in Rational Application Developer for WebSphere Software, copy the files to the ${WAS_INSTALL_ROOT}/runtimes/bi_v7/lib/ext directory.
   - In a production environment, copy the files to the ${WAS_INSTALL_ROOT}/lib/ext directory of WebSphere Application Server.

Results

The external software dependency files for JD Edwards EnterpriseOne are now part of your runtime environment.

Installing the RAR file (for modules using stand-alone adapters only)

If you chose not to embed the adapter with your module, but instead choose to make the adapter available to all deployed applications in the server instance, you need to install the adapter in the form of a RAR file to the application server. A RAR file is a Java archive (JAR) file that is used to package a resource adapter for the Java 2 Connector (J2C) architecture.

Before you begin

You must set **Deploy connector project** to **On server for use by multiple adapters** in the J2C Bean Creation and Deployment Configuration window of the J2C Bean wizard.

About this task

Installing the adapter in the form of a RAR file results in the adapter being available to all Java EE application components running in the server run time.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.

3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **Install RAR**.
6. In the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter.

   The RAR files are typically installed in the following path:
   RAD_installation_directory/ResourceAdapters/*adapter_name*/*adapter*.rar
7. Click **Next**.
8. Optional: In the Resource adapters page, change the name of the adapter and add a description.
9. Click **OK**.
10. Click **Save** in the **Messages** box at the top of the page.

What to do next

The next step is to export the module as an EAR file that you can deploy on the server.

Exporting the module as an EAR file

Using Rational Application Developer for WebSphere Software, export your module as an EAR file. By creating an EAR file, you capture all of the contents of your module in a format that can be easily deployed to WebSphere Application Server.

Before you begin

Before you can export a module as an EAR file, you must have created a module to communicate with your service. The module should be displayed in the Rational Application Developer for WebSphere Software Enterprise Explorer view.

About this task

To export the module as an EAR file, perform the following procedure.

Procedure

1. Right-click the module and select **Export**.
2. In the Select window, expand **Java EE**.
3. Select **EAR file** and click **Next**.
4. Optional: Select the correct EAR application. The EAR application is named after your module, but with “App” added to the end of the name.
5. Browse for the folder on the local file system where the EAR file will be placed.
6. Optional: To export the source files, select the **Export source files** check box. This option is provided in case you want to export the source files in addition to the EAR file. Source files include files associated with Java components, data maps, and so on.
7. Optional: To overwrite an existing file, click **Overwrite existing file**.
8. Click **Finish**.

Results

The contents of the module are exported as an EAR file.

What to do next

Install the module in the administrative console. This deploys the module to WebSphere Application Server.

Installing the EAR file

Installing the EAR file is the last step of the deployment process. When you install the EAR file on the server and run it, the adapter, which is embedded as part of the EAR file, runs as part of the installed application.

Before you begin

You must have exported your module as an EAR file before you can install it on WebSphere Application Server.

About this task

To install the EAR file, perform the following procedure. For more information about clustering adapter module applications, see the http://www.ibm.com/software/webservers/appserv/was/library/.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Applications** > **New Application** > **New Enterprise Application**.
5. Click **Browse** to locate your EAR file and click **Next**. The EAR file name is the name of the module followed by "App."
6. Optional: If you are deploying to a clustered environment, complete the following steps.
   a. On the **Step 2: Map modules to servers** window, select the module and click **Next**.
   b. Select the name of the server cluster.
   c. Click **Apply**.
7. Click **Next**. In the Summary page, verify the settings and click **Finish**.
8. Optional: If you are using an authentication alias, complete the following steps:
   a. Expand **Security** and select **Business Integration Security**.
   b. Select the authentication alias that you want to configure. You must have administrator or operator rights to change the authentication alias configurations.
   c. Optional: If it is not already specified, type the **User name**.
   d. If it is not already specified, type the **Password**.
   e. If it is not already specified, type the password again in the **Confirm Password** field.
   f. Click **OK**.

Results

The project is now deployed and the Enterprise Applications window is displayed.

What to do next

If you want to set or reset any properties or you would like to cluster adapter project applications, make those changes using the administrative console before configuring troubleshooting tools.

Deploying the module in a clustered environment

In Rational Application Developer for WebSphere Software, you can deploy the IBM WebSphere Adapter for JD Edwards EnterpriseOne in a clustered environment.

To deploy the module in a clustered environment, use any of the following approaches.

- **Embedded module:** The adapter is embedded in the application and specific to it. The adapter cannot be shared between multiple applications.
- **Node level module with embedded activation specification:** The adapter is deployed at the node level, with the activation specification created during module creation. The adapter can be shared across multiple applications.
- **Node level module with JNDI activation specification reference:** The adapter is deployed at the node level, and the application provides a JNDI reference to the activation specification. You must create the reference at the cluster scope from the administrative console, with the same JNDI name. The adapter can be shared across multiple applications.

Deploying module embedded in the application

The adapter is deployed embedded in the application and specific to it. The adapter cannot be shared between multiple applications.

Before you begin

The following steps are a necessary prerequisite to configure and deploy the module.

- Rational Application Developer for WebSphere Software version 7.5.0.0 or above.
- A clustered topology deployment environment on the WebSphere Application Server available from Rational Application Developer for WebSphere Software.
- Create a clustered topology deployment environment, as shown in the following **Gold Topology** configuration figure.
- Deploy the adapter and the adapter applications (EAR files) in the AppTarget (the target that hosts the SCA container).

About this task

To create an application with the embedded adapter, use the J2C Bean wizard.

Procedure

1. In the Service Configuration Properties window, from the **Deploy connector project** property list, select **With module for use by single application**.
2. Create the module as described in the Business process management samples for WebSphere Adapters.
3. In the **Dependencies** option for the module, after the module is created, ensure that the **Deploy with module** option is selected for the adapter.

4. If the server is not running, right-click your server in the **Servers** view and select **Start**.
5. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
6. From the **Deployment Manager Admin Console**, click **Install applications** to deploy the application.
7. On the **Step 2: Map modules to servers** window, select the module and click **Next**. For the embedded adapter option, the adapter is deployed as part of the application.
8. In the Enterprise Applications view, select the new application **<*adapter_name*>EmbeddedModuleApp**. The new application is displayed after the application is deployed at the deployment manger level.
9. Select the node and click **Installed applications** to view the deployed application on each individual node.

Results

The resource adapter is embedded and deployed as part of the application.

Deploying module at node level with embedded activation specification

The adapter is deployed at the node level, with the activation specification created during module creation. The adapter can be shared across multiple applications.

Before you begin

The following steps are a necessary prerequisite to configure and deploy the module.

- Rational Application Developer for WebSphere Software version 7.5.0.0 or above.
- A clustered topology deployment environment on the WebSphere Application Server available from Rational Application Developer for WebSphere Software.
- Create a clustered topology deployment environment, as shown in the following **Gold Topology** configuration figure.
- Deploy the adapter and the adapter applications (EAR files) in the AppTarget (the target that hosts the SCA container).

About this task

To create an application with the node level adapter and activation specification properties specified in the module itself, use the J2C Bean wizard.

Procedure

1. In the Service Configuration Properties window, from the **Deploy connector project** property list, select **On server for use by multiple applications**.
2. From the **Connection properties** list, select **Use properties below**.
3. Create the module as described in the Business process management samples for WebSphere Adapters.
4. In the **Dependencies** option for the module, ensure that the **Deploy with module** option is not selected for the adapter. Here, the adapter is not part of the module, therefore you must deploy the adapter before deploying the application.

5. If the server is not running, right-click your server in the **Servers** view and select **Start**.
6. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**. Log on to the administrative console.
7. To deploy the adapter at individual nodes, click **Resources** > **Resource Adapters** > **Resource adapters**. In the clustered environment, you must install the adapter in each node separately.
8. In the Resource adapters page, click **Install RAR**.
9. In the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter. Deploy the RAR on each node.

   The RAR files are typically installed in the following path:
   RAD_installation_directory/ResourceAdapters/*adapter_name*/*adapter*.rar
10. For deployment at node level, do not select any **Scope** because the scope is always **Node**. Click **Next**.
11. Optional: In the Resource adapters page, change the name of the adapter and add a description. Click **OK**.
12. Click **Save** in the **Messages** box at the top of the page.
13. For node level deployment, check if the adapter RAR is deployed at the node level.
14. To deploy the adapter at the cluster level, click **Resources** > **Resource Adapters** > **Resource adapters**.
15. In the Resource adapters window, set the **Scope** to **Cluster**, and then click **New**.
16. Select the RAR deployed at the node level.
17. Check if the adapter RAR is now deployed at the cluster level. Deploy the application after the adapter is deployed at the node level on the individual nodes, and then at the cluster level.
18. From the **Deployment Manager Admin Console**, click **Install applications** to deploy the application.
19. On the **Step 2: Map modules to servers** window, select the module and click **Next**. The adapter is not part of the deployed application.
20. In the **Admin Console**, click **Resources** > **Resource Adapters** > **IBM WebSphere Adapter for JD Edwards EnterpriseOne** > **J2C activation specifications** to view the activation specification from the adapter deployed at the cluster level.

Results

The resource adapter is deployed at the node level, with the activation specification.

Deploying module at node level with JNDI activation specification

The adapter is deployed at the node level, and the application provides a JNDI reference to the activation specification. You must create the activation specification with the same JNDI name at the cluster scope from the administrative console. The adapter can be shared across multiple applications

Before you begin

The following steps are a necessary prerequisite to configure and deploy the module.

- Rational Application Developer for WebSphere Software version 7.5.0.0 or above.
- A clustered topology deployment environment on the WebSphere Application Server available from Rational Application Developer for WebSphere Software.
- Create a clustered topology deployment environment, as shown in the following **Gold Topology** configuration figure.
- Deploy the adapter and the adapter applications (EAR files) in the AppTarget (the target that hosts the SCA container).

About this task

To create an application with the node level adapter and activation specification properties specified in the module itself, use the J2C Bean wizard.

Procedure

1. In the Service Configuration Properties window, from the **Deploy connector project** property list, select **On server for use by multiple applications**.
2. From the **Connection properties** list, select **Use JNDI lookup name configured on server**.
3. In the **JNDI lookup name** property field, specify the JNDI name. Use this same JNDI name when you create the activation specification from the Admin Console.
4. Create the module as described in the Business process management samples for WebSphere Adapters.
5. In the **Dependencies** option for the module, ensure that the **Deploy with module** option is not selected for the adapter.
6. If the server is not running, right-click your server in the **Servers** view and select **Start**.
7. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**. Log on to the administrative console.
8. To install the adapter at the node level, click **Resources** > **Resource Adapters** > **Resource adapters**. In the clustered environment, you must install the adapter in each node separately.
9. In the Resource adapters page, click **Install RAR**.
10. In the Install RAR file page, click **Browse** and navigate to the RAR file for your adapter. Deploy the RAR on each node.

    The RAR files are typically installed in the following path:
    IID_installation_directory/ResourceAdapters/*adapter_name*/*adapter*.rar
11. For deployment at node level, do not select any **Scope** because the scope is always **Node**. Click **Next**.
12. Optional: In the Resource adapters page, change the name of the adapter and add a description. Click **OK**.
13. Click **Save** in the **Messages** box at the top of the page.
14. To install the RAR at the cluster level, click **Resources** > **Resource Adapters** > **Resource adapters**
15. In the Resource adapters page, set the **Scope** to **Cluster**, and then click **New**.

16. Select the RAR deployed at the node level, and then check if the adapter RAR is now deployed at the cluster level. Deploy the application after the adapter is deployed at the node level on the individual nodes, and then at the cluster level.
17. From the **Deployment Manager Admin Console**, click **Install applications** to deploy the application.
18. In the **Admin Console**, click **Resources** > **Resource Adapters** > **IBM WebSphere Adapter for JD Edwards EnterpriseOne** > **J2C activation specifications** > **New** to create the activation specification from the adapter deployed at the cluster level.
19. When installing the adapter, in the **Name** field, you must enter the same name as defined in the RAR.
20. In the **JNDI name** field, you must enter the same name as given during the module creation.
21. Click **Resources** > **Resource Adapters** > **IBM WebSphere Adapter for JD Edwards EnterpriseOne** > **J2C activation specifications** to check if the JNDI reference on the adapter is same as the one specified for the module.
22. Click **Resources** > **Resource Adapters** > **IBM WebSphere Adapter for JD Edwards EnterpriseOne** > **J2C activation specifications** > **Custom properties** to set values for the activation specification in the Admin Console.
23. From the **Deployment Manager Admin console**, click **Install applications** to deploy the application after you deploy the RAR and create the activation specification.
24. On the **Step 2: Map modules to servers** page, select the module and click **Next**. The adapter is not part of the deployed application.

Results

The resource adapter is deployed at the node level, with the JNDI activation specification reference.

Chapter 5. Configuring the application on WebSphere Application Server

When you are running the adapter in a stand-alone deployment, use the administrative console of the server to start, stop, monitor, and troubleshoot the adapter module. In an application that uses an embedded adapter, the adapter module starts or stops when the application is started or stopped.

Configuring logging and tracing

Configure logging and tracing to suit your requirements. Enable logging for the adapter to control the status of event processing. Change the adapter log and trace file names to separate them from other log and trace files.

Configuring logging properties

Use the administrative console to enable logging and to set the output properties for a log, including the location, level of detail, and output format of the log.

About this task

Before the adapters can log monitored events, you must specify the service component event points that you want to monitor, what level of detail you require for each event, and format of the output used to publish the events to the logs. Use the administrative console to perform the following tasks:

- Enable or disable a particular event log
- Specify the level of detail in a log
- Specify where log files are stored and how many log files are kept
- Specify the format for log output

  If you set the output for log analyzer format, you can open trace output using the Log Analyzer tool, which is an application included with your application server. This is useful if you are trying to correlate traces from two different server processes, because it allows you to use the merge capability of the Log Analyzer.

Note: For more information about monitoring on a application server, including service components and event points, see http://publib.boulder.ibm.com/infocenter/dmndhelp/v8r0mx/topic/com.ibm.wbpm.admin.doc/topics/welcome_wps_mon.html.

You can change the log configuration statically or dynamically. Static configuration takes effect when you start or restart the application server. Dynamic or run time configuration changes apply immediately.

When a log is created, the detail level for that log is set from the configuration data. If no configuration data is available for a particular log name, the level for that log is obtained from the parent of the log. If no configuration data exists for the parent log, the parent of that log is checked, and so on, up the tree, until a log with a non-null level value is found. When you change the level of a log, the change is propagated to the child logs, which recursively propagate the change to their child log, as necessary.

To enable logging and set the output properties for a log, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, select **Servers** > **WebSphere application servers**.
2. Click the name of the server that you want to work with.
3. Under **Troubleshooting**, click **Logging and tracing**.
4. Click **Change log detail levels**.
5. Specify when you want the change to take effect:
   - For a static change to the configuration, click the **Configuration** tab.
   - For a dynamic change to the configuration, click the **Runtime** tab.
6. Click the names of the packages whose logging level you want to modify. The package names for WebSphere Adapters start with **com.ibm.j2ca.***:
   - For the adapter base component, select **com.ibm.j2ca.base.***.
   - For the adapter base component and all deployed adapters, select **com.ibm.j2ca.***.
   - For the WebSphere Adapter for JD Edwards EnterpriseOne only, select the **com.ibm.j2ca.jde.*** package.
7. Select the logging level.

Logging Level	Description
Fatal	The task cannot continue or the component cannot function.
Severe	The task cannot continue, but the component can still function. This logging level also includes conditions that indicate an impending fatal error, that is, situations that strongly suggest that resources are on the verge of being depleted.
Warning	A potential error has occurred or a severe error is impending. This logging level also includes conditions that indicate a progressive failure, for example, the potential leaking of resources.
Audit	A significant event has occurred that affects the server state or resources.
Info	The task is running. This logging level includes general information outlining the overall progress of a task.
Config	The status of a configuration is reported or a configuration change has occurred.
Detail	The subtask is running. This logging level includes general information detailing the progress of a subtask.

8. Click **Apply**.
9. Click **OK**.
10. Optional: To have static configuration changes take effect, stop and then restart the application server.

Results

Log entries from this point forward contain the specified level of information for the selected adapter components.

Changing the log and trace file names

To keep the adapter log and trace information separate from other processes, use the administrative console to change the file names. By default, log and trace information for all processes and applications on a application server is written to the `SystemOut.log` and `trace.log` files.

Before you begin

You can change the log and trace file names at any time after the adapter module has been deployed to an application server.

About this task

You can change the log and trace file names statically or dynamically. Static changes take effect when you start or restart the application server. Dynamic or run time changes apply immediately.

Log and trace files are in the *install_root*/profiles/*profile_name*/logs/*server_name* folder.

To set or change the log and trace file names, use the following procedure.

Procedure

1. In the navigation pane of the administrative console, select **Applications > Application Types > WebSphere application servers**.
2. In the Enterprise Applications list, click the name of the adapter application. This is the name of the EAR file for the adapter, but without the ear file extension. For example, if the EAR file is named `Accounting_OutboundApp.ear`, then click **Accounting_OutboundApp**.
3. In the Configuration tab, select **Modules>Manage Modules**.
4. In the list of modules, click IBM WebSphere Adapter for JD Edwards EnterpriseOne.
5. In the Configuration tab, under Additional Properties, click **Resource Adapter**.
6. In the Configuration tab, under Additional Properties, click **Custom properties**.
7. In the Custom Properties table, change the file names.
   a. Click either **logFilename** to change the name of the log file or **traceFilename** to change the name of the trace file.
   b. In the Configuration tab, type the new name in the **Value** field. By default, the log file is called `SystemOut.log` and the trace file is called `trace.log`.
   c. Click **Apply** or **OK**. Your changes are saved on your local machine.
   d. To save your changes to the master configuration on the server, use one of the following procedures:
      - **Static change**: Stop and restart the server. This method allows you to make changes, but those changes do not take effect until you stop and start the server.
      - **Dynamic change**: Click the **Save** link in the Messages box above the Custom properties table. Click **Save** again when prompted.

Changing configuration properties for embedded adapters

To change the configuration properties after you deploy the adapter as part of a module, you use the administrative console of the runtime environment. You can update resource adapter properties (used for general adapter operation) and managed connection factory properties (used for outbound processing). For information about configuring logging properties and changing the log and trace file names, see "Configuring logging and tracing" on page 57.

Setting resource adapter properties for embedded adapters

To set resource adapter properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Application Server.

About this task

Custom properties are default configuration properties shared by all IBM WebSphere Adapters.

To configure properties using the administrative console, use the following procedure:

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change. The **Configuration** page is displayed.
6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
8. From the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **Custom properties**.
10. For each property you want to change, perform the following steps.

    Note: See "Resource adapter properties" on page 86 for more information about these properties.

    a. Click the name of the property. The **Configuration** page for the selected property is displayed.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. In the Messages area, click **Save**.

Results

The resource adapter properties associated with your adapter module are changed.

Setting managed (J2C) connection factory properties for embedded adapters

With Rational Application Developer for WebSphere Software, you can work in the non-managed deployment mode for outbound operations. With this option, you can select and configure properties for your embedded adapter.

Before you begin

You must first configure managed connection factory properties for your adapters and deploy the adapter module on WebSphere Application Server.

About this task

You use managed connection factory properties to configure the target JD Edwards EnterpriseOne server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. In the Enterprise Applications list, click the name of the adapter module whose properties you want to change.
6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
8. In the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **J2C connection factories**.
10. Click the name of the connection factory associated with your adapter module.
11. In the **Additional Properties** list, click **Custom properties**.

    Custom properties are those J2C connection factory properties that are unique to IBM WebSphere Adapter for JD Edwards EnterpriseOne. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
12. For each property you want to change, perform the following steps.

    Note: See “Managed connection factory properties” on page 90 for more information about these properties.

a. Click the name of the property.
b. Change the contents of the **Value** field or type a value, if the field is empty.
c. Click **OK**.

13. In the Messages area, click **Save**.

Results

You can view the managed connection factory properties associated with your adapter module.

Setting activation specification properties for embedded adapters

To set activation specification properties for your adapter after it has been deployed as part of a module, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter module must be deployed on WebSphere Application Server.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Select **Applications** > **Application Types** > **WebSphere enterprise application**.
5. From the Enterprise Applications list, click the name of the adapter module whose properties you want to change.
6. Under **Modules**, click **Manage Modules**.
7. Click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
8. From the **Additional Properties** list, click **Resource Adapter**.
9. On the next page, from the **Additional Properties** list, click **J2C activation specifications**.
10. Click the name of the activation specification associated with the adapter module.
11. From the **Additional Properties** list, click **J2C activation specification custom properties**.
12. For each property you want to change, perform the following steps.

Note: See "Activation specification properties" on page 104 for more information about these properties.

a. Click the name of the property.
b. Change the contents of the **Value** field or type a value, if the field is empty.
c. Click **OK**.

13. In the Messages area, click **Save**.

Results

The activation specification properties associated with your adapter module are changed.

Changing configuration properties for stand-alone adapters

To set configuration properties after you install a stand-alone adapter, use the administrative console of the runtime environment. Provide the general information about the adapter and then set the resource adapter properties (which are used for general adapter operation). If the adapter is used for outbound operations, create a connection factory and then set the properties for it. For information about configuring logging properties and changing the log and trace file names, see "Configuring logging and tracing" on page 57.

Setting resource adapter properties for stand-alone adapters

To set resource adapter properties for your stand-alone adapter after it has been installed on WebSphere Application Server, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Application Server.

About this task

Custom properties are default configuration properties shared by all IBM WebSphere Adapters.

To configure properties using the administrative console, use the following procedure:

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
6. In the **Additional Properties** list, click **Custom properties**.
7. For each property you want to change, perform the following steps.

a. Click the name of the property.
b. Change the contents of the **Value** field or type a value, if the field is empty.
c. Click **OK**.

8. In the Messages area, click **Save**.

Results

The resource adapter properties associated with your adapter are changed.

Setting managed (J2C) connection factory properties for stand-alone adapters

To set managed connection factory properties for your stand-alone adapter after it has been installed on WebSphere Application Server, use the administrative console. You select the name of the property you want to configure and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Application Server.

About this task

You use managed connection factory properties to configure the target JD Edwards EnterpriseOne server instance.

Note: In the administrative console, the properties are referred to as "J2C connection factory properties."

To configure properties using the administrative console, use the following procedure:

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.
2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
6. In the **Additional Properties** list, click **J2C connection factories**.
7. If you are going to use an existing connection factory, skip ahead to select from the list of existing connection factories.

   Note: If you have selected **Specify connection properties** when you use the J2C Bean wizard to configure the adapter module, you do not need to create a connection factory.

   If you are creating a connection factory, perform the following steps:

   a. Click **New**.
   b. In the **General Properties** section of the **Configuration** tab, type a name for the connection factory. For example, you can type `AdapterCF`.

c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterCF`.
d. Optional: Select an authentication alias from the **Component-managed authentication alias** list.
e. Click **OK**.
f. In the Messages area, click **Save**.

   The newly created connection factory is displayed.

8. In the list of connection factories, click the one you want to use.
9. In the **Additional Properties** list, click **Custom properties**.

   Custom properties are those J2C connection factory properties that are unique to WebSphere Adapter for JD Edwards EnterpriseOne. Connection pool and advanced connection factory properties are properties you configure if you are developing your own adapter.
10. For each property you want to change, perform the following steps.

    Note: See "Managed connection factory properties" on page 90 for more information about these properties.
    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

The managed connection factory properties associated with your adapter are set.

Setting activation specification properties for stand-alone adapters

To set activation specification properties for your stand-alone adapter after it has been installed on WebSphere Application Server, use the administrative console. You select the name of the message endpoint property you want to configure, and then change or set the value.

Before you begin

Your adapter must be installed on WebSphere Application Server.

About this task

You use activation specification properties to configure the endpoint for inbound processing.

To configure properties using the administrative console, use the following procedure.

Procedure

1. If the server is not running, right-click your server in the **Servers** view and select **Start**.

2. When the server status changes to **Started**, right-click the server and select **Administration** > **Run administrative console**.
3. Log on to the administrative console.
4. Click **Resources** > **Resource Adapters** > **Resource adapters**.
5. In the Resource adapters page, click **IBM WebSphere Adapter for JD Edwards EnterpriseOne**.
6. In the **Additional Properties** list, click **J2C activation specifications**.
7. If you are going to use an existing activation specification, skip ahead to select from an existing list of activation specifications.

   Note: If you have selected **Use predefined connection properties** when you use the J2C Bean wizard to configure the adapter module, you must create an activation specification.

   If you are creating an activation specification, perform the following steps:
   a. Click **New**.
   b. In the **General Properties** section of the **Configuration** tab, type a name for the activation specification. For example, you can type `AdapterAS`.
   c. Type a value for **JNDI name**. For example, you can type `com/eis/AdapterAS`.
   d. Optional: Select an authentication alias from the **Authentication alias** list.
   e. Select a message listener type.
   f. Click **OK**.
   g. Click **Save** in the **Messages** box at the top of the page.

      The newly created activation specification is displayed.
8. In the list of activation specifications, click the one you want to use.
9. In the Additional Properties list, click **J2C activation specification custom properties**.
10. For each property you want to set, perform the following steps.
    a. Click the name of the property.
    b. Change the contents of the **Value** field or type a value, if the field is empty.
    c. Click **OK**.
11. After you have finished setting properties, click **Apply**.
12. In the Messages area, click **Save**.

Results

The activation specification properties associated with your adapter are set.

Adding dependency libraries to the deployed resource adapter

The deployed resource adapter running in the WebSphere Application Server requires the same dependency libraries as it does in Rational Application Developer for WebSphere Software to process requests. The method for adding these library files depends on the mode of the resource adapter deployment: standalone or embedded in the EAR file.

Stand-alone deployment

The dependency libraries may be added to the resource adapter deployed stand-alone either during initial deployment of the RAR file or by configuring the Resource Adapter properties after deployment. To set the values during initial

deployment of the RAR file, specify Class path and Native path locations. Class path is used to point to JAR files, and Native path is used to point to native libraries, such as *.dll, *.so. To set the dependency library path files after the adapter has been installed on WebSphere Application Server, use the administrative console to modify the values for the Resource Adapter.

EAR deployment

For the rare case when the connector needs to be embedded in the EAR file, the dependant libraries are added as shared libraries. Define the appropriate shared library containing software dependency files and associate them with the EAR file.

About this task

There are two methods to do this task:

- Using enhanced EAR editor in Rational Application Developer for WebSphere Software
- Using administrative console of the WebSphere Application Server

Using enhanced EAR editor:

You can use the EAR editor in Rational Application Developer for WebSphere Software to add the dependency libraries.

About this task

To create shared libraries using the EAR editor, use the following procedure.

Procedure

1. Open Enhanced EAR editor.
2. Click the **Deployment** tab.
3. Expand **Shared Library** section.
4. Click **Add** to add new shared library.
5. Specify the shared library parameters and click **OK**.
6. Deploy the EAR to the server.

Results

The dependent libraries are added as shared libraries.

Using administrative console of the WebSphere Application Server:

You can use the administrative console of the WebSphere Application Server to add the dependency libraries.

Before you begin

Ensure that the dependent files are available on the server machine in the separate folder. If needed, copy the dependent files on the server machine.

Procedure

1. Define WebSphere variables to point to appropriate folders.
2. Define the shared library through the server administrative console; you can specify it using WebSphere variables defined in step 1.

3. Deploy the EAR to the server.
4. Configure the EAR to reference defined shared library.

Results

The dependent libraries are added as shared libraries.

Chapter 6. Troubleshooting and support

Common troubleshooting techniques and self-help information help you identify and solve problems quickly. For information about configuring logging properties and changing the log and trace file names, see “Configuring logging and tracing” on page 57.

Techniques for troubleshooting problems

Troubleshooting is a systematic approach to solving a problem. The goal is to determine why something does not work as expected and how to resolve the problem. Certain common techniques can help with the task of troubleshooting.

The first step in the troubleshooting process is to describe the problem completely. Without a problem description, neither you or IBM® can know where to start to find the cause of the problem. This step includes asking yourself basic questions, such as:

- What are the symptoms of the problem?
- Where does the problem occur?
- When does the problem occur?
- Under which conditions does the problem occur?
- Can the problem be reproduced?

The answers to these questions typically lead to a good description of the problem, and that is the best way to start down the path of problem resolution.

What are the symptoms of the problem?

When starting to describe a problem, the most obvious question is "What is the problem?" Which might seem like a straightforward question; however, you can break it down into several more-focused questions that create a more descriptive picture of the problem. These questions can include:

- Who, or what, is reporting the problem?
- What are the error codes and messages?
- How does the system fail? For example, is it a loop, hang, lock up, performance degradation, or incorrect result?
- What is the business impact of the problem?

Where does the problem occur?

Determining where the problem originates is not always simple, but it is one of the most important steps in resolving a problem. Many layers of technology can exist between the reporting and failing components. Networks, disks, and drivers are only a few components to be considered when you are investigating problems.

The following questions can help you to focus on where the problem occurs in order to isolate the problem layer.

- Is the problem specific to one platform or operating system, or is it common for multiple platforms or operating systems?
- Is the current environment and configuration supported?

Remember that if one layer reports the problem, the problem does not necessarily originate in that layer. Part of identifying where a problem originates is

understanding the environment in which it exists. Take some time to completely describe the problem environment, including the operating system and version, all corresponding software and versions, and hardware information. Confirm that you are running within an environment that is a supported configuration; many problems can be traced back to incompatible levels of software that are not intended to run together or have not been fully tested together.

When does the problem occur?

Develop a detailed timeline of events leading up to a failure, especially for those cases that are one-time occurrences. You can most simply do this by working backward: Start at the time an error was reported (as precisely as possible, even down to the millisecond), and work backward through the available logs and information. Typically, you need to look only as far as the first suspicious event that you find in a diagnostic log; however, this is not always simple to do and takes practice. Knowing when to stop looking is especially difficult when multiple layers of technology are involved, and when each has its own diagnostic information.

To develop a detailed timeline of events, answer the following questions:
- Does the problem happen only at a certain time of day or night?
- How often does the problem happen?
- What sequence of events leads up to the time that the problem is reported?
- Does the problem happen after an environment change, such as upgrading or installing software or hardware?

Responding to these types of questions can provide you with a frame of reference in which to investigate the problem.

Under which conditions does the problem occur?

Knowing what other systems and applications are running at the time that a problem occurs is an important part of troubleshooting. These and other questions about your environment can help you to identify the root cause of the problem:
- Does the problem always occur when the same task is being performed?
- Does a certain sequence of events need to occur for the problem to surface?
- Do any other applications fail at the same time?

Answering these types of questions can help you explain the environment in which the problem occurs and correlate any dependencies. Remember that just because multiple problems might have occurred around the same time, the problems are not necessarily related.

Can the problem be reproduced?

From a troubleshooting standpoint, the "ideal" problem is one that can be reproduced. Typically with problems that can be reproduced, you have a larger set of tools or procedures at your disposal to help you investigate. Consequently, problems that you can reproduce are often simpler to debug and solve. However, problems that you can reproduce can have a disadvantage: If the problem is of significant business impact, you do not want it to recur! If possible, re-create the problem in a test or development environment, which typically offers you more flexibility and control during your investigation.

Tip: Simplify the scenario to isolate the problem to a suspected component.

The following questions can help you with reproducing the problem:
- Can the problem be re-created on a test machine?
- Are multiple users or applications encountering the same type of problem?
- Can the problem be re-created by running a single command, a set of commands, a particular application, or a stand-alone application?

Log and Trace Analyzer

The adapter creates log and trace files that can be viewed with the Log and Trace Analyzer.

The Log and Trace Analyzer can filter log and trace files to isolate the messages and trace information for the adapter. It can also highlight the adapter's messages and trace information in the log viewer.

The adapter's component ID for filtering and highlighting is a string composed of the characters `JDERA` plus the value of the adapter ID property. For example, if the adapter ID property is set to `001`, the component ID is `JDERA001`.

If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for JD Edwards EnterpriseOne to `001` and `002`. The component IDs for those instances, `JDERA001` and `JDERA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `JDERAInstance`.

For outbound processing, the adapter ID property is located in both the resource adapter and managed connection factory property groups. If you update the adapter ID property after using the J2C Bean wizard to configure the adapter for outbound processing, be sure to set the resource adapter and managed connection factory properties consistently. It prevents inconsistent marking of the log and trace entries. For inbound processing, the adapter ID property is located only in the resource adapter properties, so this consideration does not apply.

For more information, see the “Adapter ID (AdapterID)” on page 87 property.

First-failure data capture (FFDC) support

The adapter supports first-failure data capture (FFDC), which provides persistent records of failures and significant software incidents that occur during run time in WebSphere Application Server.

The FFDC feature runs in the background and collects events and errors that occur at run time. The feature provides a means for associating failures to one another, allowing software to link the effects of a failure to their causes, and thereby facilitate the quick location of the root cause of a failure. The data that is captured can be used to identify exception processing that occurred during the adapter run time.

When a problem occurs, the adapter writes exception messages and context data to a log file, which is located in the *install_root*/profiles/*profile*/logs/ffdc directory.

For more information about first-failure data capture (FFDC), see the WebSphere Application Server documentation.

Solutions to some common problems

Solutions and workarounds to some problems you may encounter while running WebSphere Adapter for JD Edwards EnterpriseOne with your database are provided. These problems and solutions are also documented as technotes on the Software support website.

For a complete list of technotes about WebSphere Adapters, see http://www-1.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm.

Adapter returns version conflict exception message

Problem

When you install multiple adapters with different versions of CWYBS_AdapterFoundation.jar, and if a lower version of the CWYBS_AdapterFoundation.jar is loaded during runtime, the adapter will return the ResourceAdapterInternalException error message, due to a version conflict. For example, when you install Oracle E-Business Suite adapter version 7.0.0.3 and WebSphere Adapter for JD Edwards EnterpriseOne version 7.5.0.2, the following error message is displayed "The version of CWYBS_AdapterFoundation.jar is not compatible with IBM WebSphere Adapter for JD Edwards EnterpriseOne" as IBM WebSphere Adapter for JD Edwards EnterpriseOne loads file:/C:/IBM/WebSphere/ProcServer7/profiles/ProcSrv01/installedConnectors/CWYOE_OracleEBS.rar/CWYBS_AdapterFoundation.jar with version 7.0.0.3. However, the base level of this jar required is version 7.5.0.2. To overcome this conflict, you must ensure that all adapters are at same version level. For further assistance, contact WebSphere Adapters Support for help.

Solution

Ensure that all adapters are at the same version level.

For further assistance, visit http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family.

Endpoint application of the passive adapter instance listens to the events when enableHASupport is set to True

Problem

In the active-passive configuration mode of the adapters, the endpoint application of the passive adapter instance also listens to the events or messages even if the enableHASupport property is set to `True`.

Cause

By default, in WebSphere Application Server, the `alwaysactivateAllMDBs` property in the JMS activation specification is set to `True`. This enables the endpoint application of all the adapter (active or passive) instances to listen to the events.

Solution

To stop the endpoint application of the passive adapter instance from listening to the events, you must set the `alwaysactivateAllMDBs` property value to `False`. The JMS activation specification is associated with one or more MDBs and provides the necessary configuration to receive events. If the `alwaysActivateAllMDBs` property is set to `False`, then the endpoint application of only the active adapter instance receives the events.

Perform the following procedure, to set the `alwaysActivateAllMDBs` property to `False`.

1. Log on to the administrative console.
2. Go to **Resources> JMS > Activation specifications**.
3. Click the activation specification corresponding to the application from the list.
4. Click **Custom properties** under **Additional properties**.
5. Click **alwaysActivateAllMDBs**.
6. Change the value to **False**.
7. Click **Apply** and **OK**.

Result

The endpoint application of only the active adapter instance listens to the events.

Support

This section provides information about how to troubleshoot a problem with your IBM® software, including instructions for searching knowledge bases, downloading fixes, and obtaining support.

Searching knowledge bases (Web search)

You can often find solutions to problems by searching IBM knowledge bases. You can optimize your results by using available resources, support tools, and search methods.

About this task

You can find useful information by searching the information center for Product X. However, sometimes you need to look beyond the information center to answer your questions or resolve problems.

To search knowledge bases for information that you need, use one or more of the following approaches:

- Search for content by using the IBM® Support Assistant (ISA).

  ISA is a no-charge software serviceability workbench that helps you answer questions and resolve problems with IBM software products. You can find instructions for downloading and installing ISA on the ISA website.
- Find the content that you need by using the IBM Support Portal.

  The IBM Support Portal is a unified, centralized view of all technical support tools and information for all IBM systems, software, and services. The IBM

Support Portal lets you access the IBM electronic support portfolio from one place. You can tailor the pages to focus on the information and resources that you need for problem prevention and faster problem resolution. Familiarize yourself with the IBM Support Portal by viewing the demo videos (https://www.ibm.com/blogs/SPNA/entry/the_ibm_support_portal_videos) about this tool. These videos introduce you to the IBM Support Portal, explore troubleshooting and other resources, and demonstrate how you can tailor the page by moving, adding, and deleting portlets.

- Search for content by using the IBM masthead search. You can use the IBM masthead search by typing your search string into the Search field at the top of any ibm.com® page.
- Search for content by using any external search engine, such as Google, Yahoo, or Bing. If you use an external search engine, your results are more likely to include information that is outside the ibm.com domain. However, sometimes you can find useful problem-solving information about IBM products in newsgroups, forums, and blogs that are not on ibm.com.

Tip: Include "IBM" and the name of the product in your search if you are looking for information about an IBM product.

Getting Fixes

A product fix might be available to resolve your problem.

About this task

To get product fixes, perform the following steps.

Procedure

1. Determine which fix you need. Check the list of IBM WebSphere Adapter for JD Edwards EnterpriseOne recommended fixes to confirm that your software is at the latest maintenance level. Check the list of problems fixed in the IBM WebSphere Adapter for JD Edwards EnterpriseOne fix readme documentation that is available for each listed fix pack to see if IBM has already published an individual fix to resolve your problem. To determine what fixes are available using IBM Support Assistant, run a query on fix from the search page.

   Individual fixes are published as often as necessary to resolve defects in WebSphere Application Server IBM WebSphere Adapter for JD Edwards EnterpriseOne. In addition, two kinds of cumulative collections of fixes, called fix packs and refresh packs, are published periodically for IBM WebSphere Adapter for JD Edwards EnterpriseOne, in order to bring users up to the latest maintenance level. You should install these update packages as early as possible in order to prevent problems.

   Note: A list of recommended, generally available (GA) fixes for the WebSphere Java™ Connector Architecture (JCA) and WebSphere Business Integration adapters are available here. If a Fix Pack is not available for an adapter, it implies that the GA version is the recommended version and details about that version of the adapter can be found in the Release notes.
2. Download the fix. Open the download document and follow the link in the Download package section. When downloading the file, ensure the name of the maintenance file is not changed. This includes both intentional changes and inadvertent changes caused by certain web browsers or download utilities.
3. Apply the fix. Follow the instructions in the Installation Instructions section of the download document.

4. Optional: To receive weekly notification of fixes and updates, subscribe to My Support e-mail updates.

Self-help resources

Use the resources of IBM software support to get the most current support information, obtain technical documentation, download support tools and fixes, and avoid problems with WebSphere Adapters. The self-help resources also help you diagnose problems with the adapter and provide information about how to contact IBM software support.

Support website

The WebSphere Adapters software support website at http://www-947.ibm.com/support/entry/portal/Overview/Software/WebSphere/WebSphere_Adapters_Family provides links to many resources to help you learn about, use, and troubleshoot WebSphere Adapters, including:

- Flashes (alerts about the product)
- Technical information including the product information center, manuals, IBM Redbooks®, and whitepapers
- Educational offerings
- Technotes

Recommended fixes

A list of recommended fixes you must apply is available at the following location: http://www.ibm.com/support/docview.wss?fdoc=aimadp&rs=695&uid=swg27010397.

Technotes

Technotes provide the most current documentation about WebSphere Adapter for JD Edwards EnterpriseOne, including the following topics:

- Problems and their currently available solutions
- Answers to frequently asked questions
- How to information about installing, configuring, using, and troubleshooting the adapter
- *IBM Software Support Handbook*

For a list of technotes for WebSphere Adapter for JD Edwards EnterpriseOne, see http://www-01.ibm.com/support/docview.wss?uid=swg27024044.

For a list of technotes for all adapters, see http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm.

Chapter 7. Reference information

To support you in your tasks, reference information includes details about business objects that are generated by the J2C Bean wizard and information about adapter properties, including those that support bidirectional transformation. It also includes pointers to adapter messages and related product information.

Business object information

You can determine the purpose of a business object by examining both the application-specific information within the business object definition file and the name of the business object. The application-specific information dictates what operations can be performed on the JD Edwards EnterpriseOne server. The name typically reflects the operation to be performed and the structure of the business object.

Application-specific information

Application-specific information (ASI) is metadata that specifies adapter-dependent information about how to process business objects for the adapter for JD Edwards EnterpriseOne.

When the J2C Bean wizard generates a business object, it automatically generates a business object definition, which is saved as an XSD (XML Schema Definition) file. The business object definition contains the application-specific information for that business object.

The adapter for JD Edwards EnterpriseOne uses application-specific information (ASI) to create queries for Create, Retrieve, Update, and Delete operations. ASI is generated by the J2C Bean wizard at three levels: the business-object level, the property level, and the operation level.

Application-specific information at the business-object-level

Application-specific information (ASI) at the business-object level is typically used to specify the name of the corresponding database table and to provide information necessary to perform a physical or logical delete operation. The following table describes the ASI at the business-object level.

Table 6. Application-specific information at the business-object level

Application-specific information	Description
Name	Name of operation
BSFN	List of business functions associated with the operation

Table 6. Application-specific information at the business-object level (continued)

Application-specific information	Description
AlwaysReturnResponse	Used to designate if the adapter returns a response business object for every request. If the value is set to true, the adapter always returns a response business object. If the value is set to false, an exception is generated after a JDE business function is executed. This exception is generated against the user's component. The default value is false. **Note:** For runtime exceptions, for example, if the adapter cannot establish a connection with the JD Edwards EnterpriseOne server, the exception is still generated against the user's component.

Application-specific information at the property level

Application-specific information (ASI) at the property level is typically used to specify the metadata for a property. ASI at the property level represents either child objects or an array of child objects. The following table describes the ASI of a complex property (a child) or a structure or table property (an array of child objects).

Table 7. Application-specific information at the property level

Application-specific information	Description	Possible values
Name	The business function parameter name as represented in JD Edwards EnterpriseOne	BSFNName
Type	The type of the business function parameter as it exists in JD Edwards EnterpriseOne	BSFN
IOType	The type of the business function parameter as it exists in JD Edwards EnterpriseOne	• `IN`: the parameter is mapped from the business object to the business function. • `OUT`: the parameter is mapped from the business function to the business object. • `INOUT`: the parameter is mapped both ways. • `DEFAULT`: the parameter is mapped using the default JD Edwards EnterpriseOne value. For adapter purposes, it is processed as INOUT.
RequiredType	Identifies if the parameter is required	• `YES`: the parameter is required. • `NO`: the parameter is not required. • `DEFAULT`: the parameter is using the JD Edwards EnterpriseOne value. For adapter purposes, it is processed as `NO`.
Length	The maximum possible length for the parameter value	None

Table 7. Application-specific information at the property level (continued)

Application-specific information	Description	Possible values

Reference	The xpath of the business object property that is used to obtain the value of this attribute. The xpath expression starts at the business function level	`BusinessFunctionContainer`
		`BusinessFunction1`
		`Prop1`
		`BusinessFunction2`
		`Prop2`
		
		If `BusinessFunction2/Prop2` property needs to be set with the value of `BusinessFunction1/Prop1`, the value of **Reference** for `BusinessFunction2/Prop2` needs to be set to `BusinessFunction1/Prop1`.

Application-specific information at the operation level

Application-specific information (ASI) at the operation level is used by the adapter to perform operations, such as to retrieve or update information in the JD Edwards EnterpriseOne server. The following table describes the ASI at the operation level.

Table 8. Application-specific information at the operation level

Application-specific information	Description	Value
Name	The name of the business object operation	• Create • Retrieve • Update • Delete • RetrieveAll
BSFN.Name	The name of the business functions to process	• Name • RollbackOnWarnings
BSFN.RollbackOnWarnings	Indicates if the adapter needs to rollback the current transaction when the business function returns with warnings	False (default setting)

Table 8. Application-specific information at the operation level (continued)

Application-specific information	Description	Value
RunOnError	Used to designate if the adapter should continue to process the sequential JDE business functions when a business function encounters an error while executing the business function. If the value is set to Yes, the adapter will continue to process the subsequent business functions, even if the business function fails. The error message is stored to the attribute, BSFNExecutionErrors. If the value is set to false, the adapter will stop the execution process and a rollback operation is performed. **Note:** If the ASI RunOnError is set to True for all business functions, the rollback operation is not performed. An error message for each business function that has failed is stored in the attribute BSFNExecutionErrors, against each business function in the response business object.	False (default setting)

Supported operations

An operation is the action that an adapter can perform on the JD Edwards EnterpriseOne server during outbound processing. The name of the operation typically indicates the type of action that the adapter takes, such as *create* or *update*.

The following tables defines the operations that the adapter for JD Edwards EnterpriseOne supports during outbound processing for business functions and XML Lists.

Table 9. Supported operations of business functions

Operation	Definition
Create	The top-level business object and all contained children are created.
Update	The top-level business object is modified. This operation can include adding and deleting child objects.
Delete	The top-level business object and any contained children are deleted.
Retrieve	The top-level business object and any contained children are retrieved.

Table 10. Supported operations of XML Lists

Operation	Definition
RetrieveAll	Retrieves all records from the JD Edwards EnterpriseOne server that correspond to the query values specified in the XML List. Returns a result set in the form of a container of JD Edwards EnterpriseOne query business objects, which represent the data for each row retrieved from the table.

Naming conventions

When the J2C Bean wizard generates a business object, it provides a name for the business object based on the name of the object in the JD Edwards EnterpriseOne server that it uses to build the business object.

When the J2C Bean wizard provides a name for the business object, it converts the name of the object to mixed case, which means that it removes any separators, such as spaces or underscores, and then capitalizes the first letter of each word. For example, if the J2C Bean wizard uses a JD Edwards EnterpriseOne server object called CUSTOMER_ADDRESS to generate a business object, it generates a business object called CustomerAddress.

The generated business object name can indicate the structure of the business object. However, business objects names have no semantic value to the adapter. This means that if you change the business object name, the behavior of the business object remains the same.

Important: If you choose to rename a business object, use the refactoring functionality in Rational Application Developer for WebSphere Software to ensure that you update all of the business object dependencies. For instructions on using refactoring to rename business objects, refer to the following link: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r5mx/topic/com.ibm.wbpm.wid.bpel.doc/selector/topics/trefacts.html.

Table 11. Naming conventions

Element	Naming convention	Example
Name of the business object container	<*name_of_business_object*> Container	GetEffectiveAddressContainer
Name of the business function	Name of the business function discovered by the J2C Bean wizard	GetEffectiveAddress
Name of the XML List	Name of the XML List table discovered by the J2C Bean wizard	F0116

Note: Business graph generation is optional and is supported for WebSphere Application Server only.

Outbound configuration properties

IBM WebSphere Adapter for JD Edwards EnterpriseOne has several categories of outbound connection configuration properties, which you set with the J2C Bean wizard while generating or creating objects and services. You can change the resource adapter and managed connection factory properties after you deploy the module to WebSphere Application Server using Rational Application Developer for WebSphere Software or the administrative console, but connection properties for the J2C Bean wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for JD Edwards EnterpriseOne are described in detail in tables included in each of the

configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	A required field (property) must have a value in order for the adapter to work. Sometimes the J2C Bean wizard provides a default value for required properties. Removing a default value from a required field on the J2C Bean wizard *will not change that default value*. When a required field contains no value at all, the J2C Bean wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. Sometimes a property is required only when another property has a specific value. When this is the case, the table notes this dependency. For example, • Yes, when the EventQueryType property is set to Dynamic • Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.
Default	The predefined value that is set by the J2C Bean wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table states No default value. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case-sensitive • Can be up to 40 characters in length. This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".

Row	Explanation
Globalized	If a property is globalized, it has national language support, meaning that you can set the value in your national language. Valid values are **Yes** and **No**.
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Setting connection properties for the J2C Bean wizard

J2C Java Bean connection properties establish a connection between the J2C Bean wizard of Rational Application Developer for WebSphere Software, a tool that is used to create business objects, and the JD Edwards EnterpriseOne server. The properties you configure in the J2C Bean wizard specify such things as connection configuration, bidi properties, and logging and tracing options.

Once a connection between the J2C Bean wizard and the JD Edwards EnterpriseOne server is established, the J2C Bean wizard is able to access the metadata it needs from the JD Edwards EnterpriseOne server to create business objects.

Note: Some of the properties that you set in the J2C Bean wizard are used as the initial value for the resource adapter and managed connection factory properties that you can specify at a later time in the wizard.

The external service connection properties and their purposes are described in the following table. A complete description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Important: If you set any of these connection properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

Table 12. External service connection properties for Adapter for JD Edwards EnterpriseOne

Property name	Description
"Environment" on page 84	Specifies the JD Edwards EnterpriseOne environment name
"Log file output location" on page 84	Specifies the location of the log file for external service
"Logging level" on page 84	Specifies the type error for which logging will occur during external service
"Password" on page 85	Password of the adapter user account on the JD Edwards EnterpriseOne environment
"Role" on page 85	Name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.
"User name" on page 86	Name of the adapter user account on the JD Edwards EnterpriseOne environment

Environment

This property specifies the JD Edwards EnterpriseOne environment name.

Table 13. Environment details

Required	Yes
Default	No default value
Property type	String
Usage	A JD Edwards EnterpriseOne environment is a user-defined pointer that indicates the location of data and objects on a JD Edwards EnterpriseOne server. Users can be authorized to use multiple JD Edwards EnterpriseOne environments on a single JD Edwards EnterpriseOne server.
Globalized	Yes
Bidi supported	Yes

Log file output location

This property specifies the location of the log file for external service.

Table 14. Log file output location details

Required	Yes
Default	The .metadata directory of the workspace
Property type	String
Usage	Use this directory to hold the log file that will list the errors that occur during the discovery process. The type of discovery errors for which logging occurs is controlled by the Logging level property.
Example	C:\IBM\wid6.1.0\workspace\.metadata\JDEMetadataDiscovery.log
Globalized	Yes
Bidi supported	No

Logging level

This property specifies the type error for which logging will occur during external service.

Table 15. Logging level details

Required	No

Possible values	`ALL`
	`OFF`
	`FINE`
	`FINER`
	`FINEST`
	`CONFIG`
	`INFO`
	`SEVERE`
	`WARNING`

Default	`SEVERE`
Property type	String
Usage	Use this property to tailor tracing capabilities. By specifying an error type, you are indicating that trace operations will occur only for errors of the type specified.

Table 15. Logging level details (continued)

Example	Accepting the default value of SEVERE will provide trace information on errors that fall into the SEVERE category. Severe errors mean that an operation cannot continue, though the adapter can still function. Severe errors also include error conditions that indicate an impending fatal error, that is, reporting on situations that strongly suggest that resources are on the verge of being depleted. Other error descriptions are as follows: • Fatal – Adapter cannot continue. Adapter cannot function • Warning – Potential error or impending error. This also includes conditions that indicate a progressive failure, for example, the potential leaking of resources. • Audit – Significant event affecting adapter state or resources • Info – General information outlining overall operation progress • Config – Configuration change or status • Detail – General information detailing operation progress
Globalized	Yes
Bidi supported	No

Password

This property specifies the password of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 16. Password details

Required	Yes
Default	No default value
Property type	String
Usage	Passwords are created and named by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in passwords.
Globalized	No
Bidi supported	Yes

Role

This property specifies the name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.

Table 17. Role details

Required	Yes
Default	No default value
Property type	String
Usage	Roles define what authority users have. Users can have multiple roles. A user's access to applications, forms, table columns, data sources, and so on, is based on one or more roles to which the user is assigned. Roles are created and named by the JD Edwards EnterpriseOne administrator.
Examples	• System administrator • Human resources • Accounting

Table 17. Role details (continued)

Globalized	No
Bidi supported	Yes

User name

This property specifies the name of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 18. User name details

Required	Yes
Default	No default value
Property type	String
Usage	User names are created by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in user names.
Globalized	Yes
Bidi supported	Yes

Resource adapter properties

The resource adapter properties control the general operation of the adapter. You set the resource adapter properties using the J2C Bean wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are deprecated:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following property that was specified as a resource adapter property in Version 6.2.x applies to the managed connection factory property group in Version 7.5.0.1.

- Timeout

The following table lists and describes the resource adapter properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Table 19. Resource adapter properties for Adapter for JD Edwards EnterpriseOne

Name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.

Table 19. Resource adapter properties for Adapter for JD Edwards EnterpriseOne (continued)

Name		
In the wizard	**In the administrative console**	**Description**
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
(Not available)	enableHASupport	Do not change this property.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 20. Adapter ID details

Required	Yes
Default	`001`
Property type	String
Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `JDERA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `JDERA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for JD Edwards EnterpriseOne to `001` and `002`. The component IDs for those instances, `JDERA001` and `JDERA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `JDERAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	Yes
Bidi supported	No

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of X's to prevent unauthorized disclosure of potentially sensitive data.

Table 21. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	No
Bidi supported	No

Enable high availability support (enableHASupport)

Do not change this property. It must be set to `true`.

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 22. Log file maximum size details

Required	No
Default	`0`
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as `0` or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 23. Log file name details

Required	No
Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes

Table 23. Log file name details (continued)

Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 24. Log number of files details

Required	No
Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 25. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 26. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 27. Trace number of files details

Required	No
Default	1
Property type	Integer
Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Managed connection factory properties

Managed connection factory properties are used by the adapter at run time to create an outbound connection instance with the JD Edwards EnterpriseOne server.

You set the managed connection factory properties using either the J2C Bean wizard or the administrative console (after deployment).

The following table lists and describes the managed connection factory properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Note: The J2C Bean wizard refers to these properties as managed connection factory properties and WebSphere Application Server administrative console refers to these as (J2C) connection factory properties.

Table 28. Managed connection factory properties for Adapter for JD Edwards EnterpriseOne

Property name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
"Environment" on page 92	environment	Specifies the JD Edwards EnterpriseOne environment name
"Password" on page 92	password	Password of the adapter user account on the JD Edwards EnterpriseOne environment
"Role" on page 93	role	Name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.
"Timeout" on page 93	timeout	This property is the global timeout value, in milliseconds, set on the XML List request execute call.
"User name" on page 93	userName	Name of the adapter user account on the JD Edwards EnterpriseOne environment
"Maximum retries on connection failure (connectionRetryLimit) " on page 94	connectionRetryLimit	Specifies the maximum number of times the adapter attempts to connect to the Enterprise Information System (EIS) to reestablish the connection.

Table 28. Managed connection factory properties for Adapter for JD Edwards EnterpriseOne (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
"Retry interval if connection fails (connectionRetryInterval)" on page 94	connectionRetryInterval	Specifies the time interval between the attempts to reconnect to EIS, if the connection fails.

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 29. Adapter ID details

Required	Yes
Default	`001`
Property type	String
Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `JDERA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `JDERA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for JD Edwards EnterpriseOne to `001` and `002`. The component IDs for those instances, `JDERA001` and `JDERA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `JDERAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	Yes
Bidi supported	No

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of `X`'s to prevent unauthorized disclosure of potentially sensitive data.

Table 30. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	No
Bidi supported	No

Environment

This property specifies the JD Edwards EnterpriseOne environment name.

Table 31. Environment details

Required	Yes
Default	No default value
Property type	String
Usage	A JD Edwards EnterpriseOne environment is a user-defined pointer that indicates the location of data and objects on a JD Edwards EnterpriseOne server. Users can be authorized to use multiple JD Edwards EnterpriseOne environments on a single JD Edwards EnterpriseOne server.
Example	
Globalized	Yes
Bidi supported	Yes

Password

This property specifies the password of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 32. Password details

Required	Yes
Default	No default value
Property type	String
Usage	Passwords are created and named by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in passwords.
Example	
Globalized	No
Bidi supported	Yes

Role

This property specifies the name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.

Table 33. Role details

Required	Yes
Default	No default value
Property type	String
Usage	Roles define what authority users have. Users can have multiple roles. A user's access to applications, forms, table columns, data sources, and so on, is based on one or more roles to which the user is assigned. Roles are created and named by the JD Edwards EnterpriseOne administrator.
Examples	• System administrator • Human resources • Accounting
Globalized	No
Bidi supported	Yes

Timeout

This property specifies the timeout value, in milliseconds, set on the XML List request call.

Table 34. Timeout details

Required	Yes
Default	30,000
Unit of measure	Milliseconds
Property type	Integer
Usage	Use the Timeout property to specify the amount of time the adapter should take to perform a RetrieveAll operation using an XML List. If no value is specified, the adapter will time out after 30 seconds (30,000 milliseconds).
Globalized	Yes
Bidi supported	No

User name

This property specifies the name of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 35. User name details

Required	Yes
Default	No default value
Property type	String
Usage	User names are created by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in user names.
Example	

Table 35. User name details (continued)

Globalized	Yes
Bidi supported	Yes

Maximum retries on connection failure (connectionRetryLimit)

This property specifies the maximum number of times the adapter will attempt to reestablish a connection to the EIS server, when the adapter encounters an error related to the outbound connection.

Table 36. Maximum retries on connection failure property characteristics

Required	No
Possible values	Integers equal to and greater than zero
Default	0
Property type	Integer
Usage	When this property is set to:0 • The adapter does not attempt to connect to the EIS server, if an error occurs during startup and or while establishing a connection. The adapter does not verify if the connection to the EIS server is valid when there is an outbound request during runtime. >0 • The adapter attempts to connect to the EIS server for the specified number of times, if an error occurs during startup or while establishing a connection. • The adapter verifies if the connection to the EIS server is valid when there is an outbound request during runtime. If the connection is not valid, it is terminated and a new connection is created to process the request. If the adapter fails to establish a connection after trying for the specified number of times, a connection error is generated. If the adapter is successful in re-establishing the connection, the outbound operation is completed.
Globalized	No
Bidi supported	No

Retry interval if connection fails (connectionRetryInterval)

This property specifies the time interval between the attempts to reconnect to EIS server if the connection fails.

Table 37. Retry interval if connection fails property characteristics

Required	No
Possible values	Integers equal to and greater than zero
Default	60000
Property type	Milliseconds
Usage	This property is applicable only if the value of the property “Maximum retries on connection failure” is set to greater than 0. When the adapter encounters an error while establishing a connection to the EIS server, this property specifies the time interval the adapter waits between attempts to reestablish a connection.
Globalized	No

Table 37. Retry interval if connection fails property characteristics (continued)

Bidi supported	No

Interaction specification properties

Interaction specification properties control the interaction for an operation. The J2C Bean wizard sets the interaction specification properties when you configure the adapter. Typically, you do not need to change these properties. However, some properties for outbound operations can be changed by the user.

One reason to change the interaction specification properties is to increase the value of the property that specifies the maximum number of records to be returned by a RetrieveAll operation, if your RetrieveAll operations do not return complete information. To change these properties after the application is deployed, use the J2C bean generated in the WebSphere Application Server administrative console. The properties reside in the method binding of the import.

The following tables list and describe the interaction specification property that you can set. For information about how to read the property details table, see "Guide to information about properties" on page 81.

Table 38. Interaction specification property for the Adapter for JD Edwards EnterpriseOne

Property name		
In the wizard		**Description**
Maximum records for RetrieveAll operation		Maximum number of records to return during a RetrieveAll operation

Maximum number of records for RetrieveAll operation

This property specifies the maximum number of records to return for a RetrieveAll operation.

Table 39. Maximum records for RetrieveAll operation details

Required	Yes
Default	100
Usage	If the number of hits in the database exceeds the value of the Maximum number of records property, the adapter returns the error MatchesExceededLimitException and MatchesExceededLimitFault. Use this property to avoid out-of-memory issues.
Property type	Integer
Globalized	No
Bidi supported	No

Inbound configuration properties

WebSphere Adapter for JD Edwards EnterpriseOne has several categories of inbound connection configuration properties, which you set with the J2C Bean wizard while generating or creating objects and services. You can change the resource adapter and activation specification properties after you deploy the module using Rational Application Developer for WebSphere Software or the administrative console, but connection properties for the J2C Bean wizard cannot be changed after deployment.

Guide to information about properties

The properties used to configure WebSphere Adapter for JD Edwards EnterpriseOne are described in detail in tables included in each of the configuration properties topics, such as Resource adapter properties, Managed connection factory properties, and so on. To help you use these tables, information about each row you might see is explained here.

The following table explains the meaning of each row that might be displayed in the table for a configuration property.

Row	Explanation
Required	A required field (property) must have a value in order for the adapter to work. Sometimes the J2C Bean wizard provides a default value for required properties. Removing a default value from a required field on the J2C Bean wizard *will not change that default value*. When a required field contains no value at all, the J2C Bean wizard processes the field using its assigned default value, and that default value is displayed on the administrative console. Possible values are **Yes** and **No**. Sometimes a property is required only when another property has a specific value. When this is the case, the table notes this dependency. For example, • Yes, when the EventQueryType property is set to Dynamic • Yes, for Oracle databases
Possible values	Lists and describes the possible values that you can select for the property.
Default	The predefined value that is set by the J2C Bean wizard. When the property is required, you must either accept the default value or specify one yourself. If a property has no default value, the table states No default value. The word `None` is an acceptable default value, and does not mean that there is no default value.
Unit of measure	Specifies how the property is measured, for example in kilobytes or seconds.
Property type	Describes the property type. Valid property types include: • Boolean • String • Integer
Usage	Describes usage conditions or restrictions that might apply to the property. For instance, here is how a restriction would be documented: For Rational Application Developer for WebSphere Software version 6.40 or earlier, the password: • Must be uppercase • Must be 8 characters in length For versions of Rational Application Developer for WebSphere Software later than 6.40, the password: • Is not case-sensitive • Can be up to 40 characters in length. This section lists other properties that affect this property or the properties that are affected by this property and describes the nature of the conditional relationship.

Row	Explanation
Example	Provides sample property values, for example: "If Language is set to JA (Japanese), code page number is set to `8000`".
Globalized	If a property is globalized, it has national language support, meaning that you can set the value in your national language. Valid values are **Yes** and **No**.
Bidi supported	Indicates whether the property is supported in bidirectional (bidi) processing. Bidirectional processing refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file. Valid values are **Yes** and **No**.

Setting connection properties for the J2C Bean wizard

J2C Java Bean connection properties establish a connection between the J2C Bean wizard of Rational Application Developer for WebSphere Software, a tool that is used to create business objects, and the JD Edwards EnterpriseOne server. The properties you configure in the J2C Bean wizard specify such things as connection configuration, bidi properties, and logging and tracing options.

Once a connection between the J2C Bean wizard and the JD Edwards EnterpriseOne server is established, the J2C Bean wizard is able to access the metadata it needs from the JD Edwards EnterpriseOne server to create business objects.

Note: Some of the properties that you set in the J2C Bean wizard are used as the initial value for the resource adapter and managed connection factory properties that you can specify at a later time in the wizard.

The external service connection properties and their purposes are described in the following table. A complete description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Important: If you set any of these connection properties using bidirectional script, you must set values that identify the format of the bidirectional script entered for that property.

Table 40. External service connection properties for Adapter for JD Edwards EnterpriseOne

Property name	Description
"Environment" on page 98	Specifies the JD Edwards EnterpriseOne environment name
"Log file output location" on page 98	Specifies the location of the log file for external service
"Logging level" on page 98	Specifies the type error for which logging will occur during external service
"Password" on page 99	Password of the adapter user account on the JD Edwards EnterpriseOne environment
"Role" on page 99	Name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.
"User name" on page 100	Name of the adapter user account on the JD Edwards EnterpriseOne environment

Environment

This property specifies the JD Edwards EnterpriseOne environment name.

Table 41. Environment details

Required	Yes
Default	No default value
Property type	String
Usage	A JD Edwards EnterpriseOne environment is a user-defined pointer that indicates the location of data and objects on a JD Edwards EnterpriseOne server. Users can be authorized to use multiple JD Edwards EnterpriseOne environments on a single JD Edwards EnterpriseOne server.
Globalized	Yes
Bidi supported	Yes

Log file output location

This property specifies the location of the log file for external service.

Table 42. Log file output location details

Required	Yes
Default	The .metadata directory of the workspace
Property type	String
Usage	Use this directory to hold the log file that will list the errors that occur during the discovery process. The type of discovery errors for which logging occurs is controlled by the Logging level property.
Example	C:\IBM\wid6.1.0\workspace\.metadata\JDEMetadataDiscovery.log
Globalized	Yes
Bidi supported	No

Logging level

This property specifies the type error for which logging will occur during external service.

Table 43. Logging level details

Required	No

Possible values	`ALL`
	`OFF`
	`FINE`
	`FINER`
	`FINEST`
	`CONFIG`
	`INFO`
	`SEVERE`
	`WARNING`

Default	`SEVERE`
Property type	String
Usage	Use this property to tailor tracing capabilities. By specifying an error type, you are indicating that trace operations will occur only for errors of the type specified.

Table 43. Logging level details (continued)

Example	Accepting the default value of SEVERE will provide trace information on errors that fall into the SEVERE category. Severe errors mean that an operation cannot continue, though the adapter can still function. Severe errors also include error conditions that indicate an impending fatal error, that is, reporting on situations that strongly suggest that resources are on the verge of being depleted. Other error descriptions are as follows: • Fatal – Adapter cannot continue. Adapter cannot function • Warning – Potential error or impending error. This also includes conditions that indicate a progressive failure, for example, the potential leaking of resources. • Audit – Significant event affecting adapter state or resources • Info – General information outlining overall operation progress • Config – Configuration change or status • Detail – General information detailing operation progress
Globalized	Yes
Bidi supported	No

Password

This property specifies the password of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 44. Password details

Required	Yes
Default	No default value
Property type	String
Usage	Passwords are created and named by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in passwords.
Globalized	No
Bidi supported	Yes

Role

This property specifies the name of the role that is associated with the user name used to access the JD Edwards EnterpriseOne environment.

Table 45. Role details

Required	Yes
Default	No default value
Property type	String
Usage	Roles define what authority users have. Users can have multiple roles. A user's access to applications, forms, table columns, data sources, and so on, is based on one or more roles to which the user is assigned. Roles are created and named by the JD Edwards EnterpriseOne administrator.
Examples	• System administrator • Human resources • Accounting

Table 45. Role details (continued)

Globalized	No
Bidi supported	Yes

User name

This property specifies the name of the adapter user account on the JD Edwards EnterpriseOne environment.

Table 46. User name details

Required	Yes
Default	No default value
Property type	String
Usage	User names are created by the JD Edwards EnterpriseOne administrator. There are no restrictions on the type of characters used, the number of characters used, or the case of the characters used in user names.
Globalized	Yes
Bidi supported	Yes

Resource adapter properties

The resource adapter properties control the general operation of the adapter. You set the resource adapter properties using the J2C Bean wizard when you configure the adapter. After deploying the adapter, use the administrative console to change these properties.

The following properties for logging and tracing are deprecated:

- LogFileMaxSize
- LogFileName
- LogNumberOfFiles
- TraceFileMaxSize
- TraceFileName
- TraceNumberOfFiles

The following property that was specified as a resource adapter property in Version 6.2.x applies to the managed connection factory property group in Version 7.5.0.1.

- Timeout

The following table lists and describes the resource adapter properties. A more detailed description of each property is provided in the sections that follow the table. For information on how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Table 47. Resource adapter properties for Adapter for JD Edwards EnterpriseOne

Name		
In the wizard	**In the administrative console**	**Description**
Adapter ID	AdapterID	Identifies the adapter instance for PMI events and for logging and tracing.

Table 47. Resource adapter properties for Adapter for JD Edwards EnterpriseOne (continued)

Name		
In the wizard	**In the administrative console**	**Description**
Disguise user data as "XXX" in log and trace files	HideConfidentialTrace	Specifies whether to disguise potentially sensitive information by writing `X` strings instead of user data in the log and trace files.
(Not available)	enableHASupport	Do not change this property.
(Not available)	LogFileSize	Deprecated
(Not available)	LogFilename	Deprecated
(Not available)	LogNumberOfFiles	Deprecated
(Not available)	TraceFileSize	Deprecated
(Not available)	TraceFileName	Deprecated
(Not available)	TraceNumberOfFiles	Deprecated

Adapter ID (AdapterID)

This property identifies a specific deployment or instance of the adapter.

Table 48. Adapter ID details

Required	Yes
Default	`001`
Property type	String
Usage	This property identifies the adapter instance in the log and trace files, and also helps identify the adapter instance while monitoring adapters. The adapter ID is used with an adapter-specific identifier, `JDERA`, to form the component name used by the Log and Trace Analyzer tool. For example, if the adapter ID property is set to `001`, the component ID is `JDERA001`. If you run multiple instances of the same adapter, ensure that the first eight characters of the adapter ID property are unique for each instance so that you can correlate the log and trace information to a particular adapter instance. By making the first seven characters of an adapter ID property unique, the component ID for multiple instances of that adapter is also unique, allowing you to correlate the log and trace information to a particular instance of an adapter. For example, when you set the adapter ID property of two instances of WebSphere Adapter for JD Edwards EnterpriseOne to `001` and `002`. The component IDs for those instances, `JDERA001` and `JDERA002`, are short enough to remain unique, enabling you to distinguish them as separate adapter instances. However, instances with longer adapter ID properties cannot be distinguished from each other. If you set the adapter ID properties of two instances to `Instance01` and `Instance02`, you will not be able to examine the log and trace information for each adapter instance because the component ID for both instances is truncated to `JDERAInstance`. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	Yes
Bidi supported	No

Disguise user data as "XXX" in log and trace files (HideConfidentialTrace)

This property specifies whether to replace user data in log and trace files with a string of X's to prevent unauthorized disclosure of potentially sensitive data.

Table 49. Disguise user data as "XXX" in log and trace files details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If you set this property to `True`, the adapter replaces user data with a string of `X`'s when writing to log and trace files. For inbound processing, the value of this property is set at the resource adapter level. For outbound processing, the value can be set both at the resource adapter level and the managed connection factory level. After you use the J2C Bean wizard to configure the adapter for outbound processing, you can set the resource adapter and managed connection factory properties independently.
Globalized	No
Bidi supported	No

Enable high availability support (enableHASupport)

Do not change this property. It must be set to `true`.

Log file maximum size (LogFileMaxSize)

This property specifies the size of the log files in kilobytes.

Table 50. Log file maximum size details

Required	No
Default	`0`
Property type	Integer
Usage	When the log file reaches it maximum size, the adapter start using a new log file. If the file size is specified as `0` or no maximum size is specified, the file does not have a maximum size.
Globalized	Yes
Bidi supported	No

Log file name (LogFilename)

This property specifies the full path name of the log file.

Table 51. Log file name details

Required	No
Default	No default value
Property type	String
Usage	This property is deprecated.
Globalized	Yes

Table 51. Log file name details (continued)

Bidi supported	Yes

Log number of files (LogNumberOfFiles)

This property specifies the number of log files.

Table 52. Log number of files details

Required	No
Default	1
Property type	Integer
Usage	When a log file reaches its maximum size, the adapter starts using another log file. If no value is specified, the adapter creates a single log file.
Globalized	Yes
Bidi supported	No

Trace file maximum size (TraceFileMaxSize)

This property specifies the size of the trace files in kilobytes.

Table 53. Trace file maximum size details

Required	No
Default	0
Property type	Integer
Usage	If no value is specified, then the trace file has no maximum size.
Globalized	Yes
Bidi supported	No

Trace file name (TraceFilename)

This property specifies the full path of the trace file.

Table 54. Trace file name details

Required	No
Default	No default value
Unit of measure	Kilobytes
Property type	String
Usage	This property is deprecated.
Globalized	Yes
Bidi supported	Yes

Trace number of files (TraceNumberOfFiles)

This property specifies the number of trace files to use. When a trace file reaches its maximum size, the adapter starts using another trace file.

Table 55. Trace number of files details

Required	No
Default	1
Property type	Integer
Usage	If no value is specified, the adapter uses a single trace file.
Globalized	Yes
Bidi supported	No

Activation specification properties

Activation specification properties hold the inbound event processing configuration information for a message endpoint. You can set activation specification properties using either the J2C Bean wizard or the administrative console.

Activation specification properties are used during endpoint activation to notify the adapter of eligible event listeners. During inbound processing, the adapter uses these event listeners to receive events before forwarding them to the endpoint.

The following table lists the activation specification properties for inbound communication. You set the activation specification properties using the J2C Bean wizard and can change them using the JNDI, in theWebSphere Application Server administrative console. A more detailed description of each property is provided in the sections that follow the table. For information about how to read the property detail tables in the sections that follow, see "Guide to information about properties" on page 81.

Table 56. Activation specification properties

Property name		
In the wizard	**In the administrative console**	**Description**
Auto Acknowledge	AutoAcknowledge	Specifies the event acknowledge mode that is used.
Delivery type	DeliveryType	Determines the order in which events are delivered by the adapter to the export..
Ensure once-only event delivery	AssuredOnceDelivery	Specifies whether the adapter provides assured once delivery of events..
Retry limit for failed events	FailedEventRetryLimit	The number of times the adapter attempts to redeliver an event before marking the event as failed.
Interval between polling periods	PollPeriod	The length of time that the adapter waits between polling periods..
Maximum connections	MaximumConnections	The maximum number of connections that the adapter can use for inbound event delivery..
Maximum events in polling period	PollQuantity	The number of events the adapter delivers to the export during each poll period..
Minimum connections	MinimumConnections	The minimum number of connections that the adapter can use for inbound event delivery..
No Wait	NoWait	Specifies whether the adapter waits for a time interval to get an event from the JD Edwards EnterpriseOne transaction server by invoking the Dynamic Java Connector API.

Table 56. Activation specification properties (continued)

Property name		
In the wizard	**In the administrative console**	**Description**
Maximum number of retries in case of system connection failure	RetryLimit	The number of times the adapter tries to reestablish an inbound connection after an error..
Retry EIS connection on startup	RetryConnectionOnStartup	Controls whether the adapter retries the connection to the JD Edwards EnterpriseOne server if it cannot connect at startup.
Time between retries in case of system connection failure (milliseconds)	RetryInterval	The length of time that the adapter waits between attempts to reestablish connection after an error during inbound operations..
Stop the adapter when an error is encountered while polling	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling..
Wait Time	WaitTime	Specifies the waiting time if the No Wait property is false.
"Stop the adapter when an error is encountered while polling (StopPollingOnError)" on page 111	StopPollingOnError	Specifies whether the adapter stops polling for events when it encounters an error during polling

Auto acknowledge (AutoAcknowledge)

This property specifies the event acknowledge mode that is used. You can specify either the auto acknowledge mode or the client acknowledge mode.

Table 57. Auto acknowledge details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	Specifies the event acknowledge mode, which is either the auto acknowledge mode or the client acknowledge mode.
Example	False
Globalized	No
Bidi supported	Yes

Delivery type (DeliveryType)

This property specifies the order in which events are delivered by the adapter to the export.

Table 58. Delivery type details

Required	No
Possible values	`ORDERED` `UNORDERED`

Table 58. Delivery type details (continued)

Default	ORDERED
Property type	String
Usage	The following values are supported: • ORDERED: The adapter delivers events to the export one at a time. • UNORDERED: The adapter delivers all events to the export at once.
Globalized	No
Bidi supported	No

Ensure once-only event delivery (AssuredOnceDelivery)

This property specifies whether to provide ensure once-only event delivery for inbound events.

Table 59. Ensure once-only event delivery details

Required	Yes
Possible values	True False
Default	True
Property type	Boolean
Usage	When this property is set to True, the adapter provides assured once event delivery. This means that each event is delivered once and only once. A value of False does not provide assured once event delivery, but provides better performance. When this property is set to True, the adapter attempts to store transaction (XID) information in the event store. If it is set to False, the adapter does not attempt to store the information. This property is used only if the export component is transactional. If it is not, no transaction can be used, regardless of the value of this property.
Globalized	No
Bidi supported	No

Retry limit for failed events (FailedEventRetryLimit)

This property specifies the number of times that the adapter attempts to redeliver an event before marking the event as failed.

Table 60. Retry limit for failed events details

Required	No
Possible values	Integers
Default	5
Property type	Integer

Table 60. Retry limit for failed events details (continued)

Usage	Use this property to control how many times the adapter tries to send an event before marking it as failed. It accepts the following values: **Default** If this property is not set, the adapter tries five additional times before marking the event as failed. **0** The adapter tries to deliver the event an infinite number of times. When the property is set to `0`, the event remains in the event store and the event is never marked as failed. **> 0** For integers greater than zero, the adapter retries the specified number of times before marking the event as failed. **< 0** For negative integers, the adapter does not retry failed events.
Globalized	No
Bidi supported	No

Interval between polling periods (PollPeriod)

This property specifies the length of time that the adapter waits between polling periods.

Table 61. Interval between polling periods details

Required	Yes
Possible values	Integers greater than or equal to `0`.
Default	`2000`
Unit of measure	Milliseconds
Property type	Integer
Usage	The poll period is established at a fixed rate, which means that if running the poll cycle is delayed for any reason (for example, if a prior poll cycle takes longer than expected to complete) the next poll cycle occurs immediately to make up for the lost time caused by the delay.
Globalized	No
Bidi supported	No

Maximum connections (MaximumConnections)

This property specifies the maximum number of connections that the adapter can use for inbound event delivery.

Table 62. Maximum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter.
Globalized	No
Bidi supported	No

Maximum events in polling period (PollQuantity)

This property specifies the number of events that the adapter delivers to the export during each poll period.

Table 63. Maximum events in polling period details

Required	Yes
Default	`10`
Property type	Integer
Usage	The value must be greater than 0. If this value is increased, more events are processed per polling period and the adapter may perform less efficiently. If this value is decreased, fewer events are processed per polling period and the adapter's performance might improve slightly.
Globalized	No
Bidi supported	No

Minimum connections (MinimumConnections)

This property specifies the minimum number of connections that the adapter can use for inbound event delivery.

Table 64. Minimum connections details

Required	No
Default	1
Property type	Integer
Usage	Only positive values are valid. Any value less than 1 is treated as 1 by the adapter.
Globalized	No
Bidi supported	No

No wait (NoWait)

This property specifies whether the adapter waits for a time interval to get an event from the JD Edwards EnterpriseOne transaction server by invoking the Dynamic Java Connector API.

Table 65. No wait details

Required	No
Possible values	`True` `False`
Default	`True`
Property type	Boolean
Usage	Specifies whether the adapter waits for a time interval to get an event from the JD Edwards EnterpriseOne transaction server by invoking the Dynamic Java Connector API.
Example	True
Globalized	No
Bidi supported	Yes

Maximum number of retries in case of system connection failure (RetryLimit)

This property specifies the number of times the adapter tries to reestablish an inbound connection.

Table 66. Maximum number of retries in case of system connection failure

Required	No
Possible values	`0` and positive integers
Default	`0`
Property type	Integer
Usage	This property controls how many times the adapter retries the connection if the adapter cannot connect to the JD Edwards EnterpriseOne server to perform inbound processing. A value of `0` indicates an infinite number of retries. To control whether the adapter retries if it cannot connect to the JD Edwards EnterpriseOne server when it is first started, use the RetryConnectionOnStartup property.
Globalized	No
Bidi supported	No

Retry EIS connection on startup (RetryConnectionOnStartup)

This property controls whether the adapter attempts to connect again to the JD Edwards EnterpriseOne server if it cannot connect at startup.

Table 67. Retry EIS connection on startup details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	This property indicates whether the adapter should retry the connection to the JD Edwards EnterpriseOne server if the connection cannot be made when the adapter is started: • Set the property to `False` when you want immediate feedback about whether the adapter can establish a connection to the JD Edwards EnterpriseOne server, for example, when you are building and testing the application that receives events from the adapter. If the adapter cannot connect, the adapter writes log and trace information and stops. The administrative console shows the application status as `Stopped`. After you resolve the connection problem, start the adapter manually. • Set the property to `True` if you do not need immediate feedback about the connection. If the adapter cannot connect during startup, it writes log and trace information, and then attempts to reconnect, using the RetryInterval property to determine how frequently to retry and the value of the RetryLimit property to retry multiple times until that value is reached. The administrative console shows the application status as `Started`.
Globalized	No
Bidi supported	No

Time between retries in case of system connection failure (RetryInterval)

When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to reestablish a connection.

Table 68. Retry interval details

Required	Yes
Default	`2000`
Unit of measure	Milliseconds
Property type	Integer
Usage	Only positive values are valid. When the adapter encounters an error related to the inbound connection, this property specifies the length of time the adapter waits before trying to establish a new connection.
Globalized	No
Bidi supported	No

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 69. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	`True` `False`
Default	`False`
Property type	Boolean
Usage	If this property is set to `True`, the adapter stops polling when it encounters an error. If this property is set to `False`, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Wait time (WaitTime)

This property specifies the waiting time if the No Wait property is false.

Table 70. Wait time details

Required	No
Possible values	Any positive integer. Any negative integer will be treated as the default value (`3000 milliseconds`)
Default	`3000`
Unit of measure	Millisecond
Property type	Integer
Usage	This property specifies the waiting time if the No Wait property is false.

Table 70. Wait time details (continued)

Example	5000
Globalized	No
Bidi supported	Yes

Stop the adapter when an error is encountered while polling (StopPollingOnError)

This property specifies whether the adapter will stop polling for events when it encounters an error during polling.

Table 71. Stop the adapter when an error is encountered while polling details

Required	No
Possible values	True
Default	`False`
Property type	Boolean
Usage	If this property is set to True, the adapter stops polling when it encounters an error. If this property is set to False, the adapter logs an exception when it encounters an error during polling and continues polling.
Globalized	No
Bidi supported	No

Globalization

WebSphere Adapter for JD Edwards EnterpriseOne is a globalized application that can be used in multiple linguistic and cultural environments. Based on character set support and the locale of the host server, the adapter delivers message text in the appropriate language. The adapter supports bidirectional script data transformation between integration components.

Globalization and bidirectional data transformation

The adapter is globalized to support single- and multi-byte character sets and deliver message text in the specified language. The adapter also performs bidirectional script data transformation, which refers to the task of processing data that contains both right-to-left (Hebrew or Arabic, for example) and left-to-right (a URL or file path, for example) semantic content within the same file.

Globalization

Globalized software applications are designed and developed for use within multiple linguistic and cultural environments rather than a single environment. WebSphere Adapters, Rational Application Developer for WebSphere Software, and WebSphere Application Server are written in Java. The Java run time environment within the Java virtual machine (JVM) represents data in the Unicode character code set. Unicode contains encodings for characters in most known character code sets (both single- and multi-byte). Therefore, when data is transferred between these integration system components, there is no need for character conversion.

To log error and informational messages in the appropriate language and for the appropriate country or region, the adapter uses the locale of the system on which it is running.

Bidirectional script data transformation

Languages such as Arabic and Hebrew are written from right to left, yet they contain embedded segments of text that are written left to right, resulting in bidirectional script. When software applications handle bidirectional script data, standards are used to display and process it. Bidirectional script data transformation applies only to string type data. useWebSphere Application Server use the Windows standard format, but applications or file systems that exchange data with the server might use a different format. The adapter transforms bidirectional script data passed between the two systems so that it is accurately processed and displayed on both sides of a transaction. It transforms the script data by using a set of properties that defines the format of script data, as well as properties that identify content or metadata to which transformation applies.

Bidirectional script data formats

WebSphere Application Server use the bidirectional format of ILYNN (implicit, left-to-right, on, off, nominal). These five attributes comprise the format used by Windows. If an application or file system that sends or receives data from the server uses a different format, the adapter converts the format prior to introducing the data to the server. For the conversion to occur, you use the J2C Bean wizard to set attribute values that represent the bidirectional format used by the sending application or file system. This is done when you deploy the adapter for the first time.

The bidirectional format consists of five attributes. When you set bidirectional properties, you assign values for each of these attributes. The attributes and settings are listed in the following table.

Table 72. Bidirectional format attributes

Letter position	Purpose	Values	Description	Default setting
1	Order schema	I	Implicit (Logical)	I
		V	Visual	
2	Direction	L	Left-to-Right	L
		R	Right-to-Left	
		C	Contextual Left-to-Right	
		D	Contextual Right-to-Left	
3	Symmetric Swapping	Y	Symmetric swapping is on	Y
		N	Symmetric swapping is off	
4	Text Shaping	S	Text is shaped	N
		N	Text is not shaped (Nominal)	
		I	Initial shaping	
		M	Middle shaping	
		F	Final shaping	
		B	Isolated shaping	

Table 72. Bidirectional format attributes (continued)

Letter position	Purpose	Values	Description	Default setting
5	Numeric Shaping	H	National (Hindi)	N
		C	Contextual shaping	
		N	Numbers are not shaped (Nominal)	

Bidirectional properties that identify data for transformation

To identify business data subject to transformation, set the BiDiContextEIS property. Do this by specifying values for each of the five bidirectional format attributes (listed in the preceding table) for the property. The BiDiContextEIS property can be set for the managed connection factory and the activation specification.

To identify application-specific data for transformation, annotate the BiDiContextEIS property and the BiDiMetadata property within a business object. Do this by using the business object editor within to Rational Application Developer for WebSphere Software to add the properties as application-specific elements of a business object.

Properties enabled for bidirectional data transformation

Bidirectional data transformation properties enforce the correct format of bidirectional script data exchanged between an application and integration tools and runtime environments. Once these properties are set, bidirectional script data is correctly processed and displayed in Rational Application Developer for WebSphere Software and WebSphere Application Server.

Managed connection properties

The following managed connection properties control bidirectional script data transformation.

- Username
- Password
- Environment
- Role

Activation specification properties

The following activation specification properties are enabled for bidirectional script data transformation:

- Auto acknowledge
- Guaranteed event delivery
- No wait
- Wait time

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 2Z4A/SOM1
294 Route 100
Somers, NY 10589-0100
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: (c) (your company name) (year). Portions of

this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning:

Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A complete and current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (http://www.eclipse.org).

Index

Special characters

A

B

C

D

E

F

G

H

I

W

X

Printed in USA