

codeobjects



CORADS

CodeObjects Rapid Application Development Studio

Getting
Started

CodeObjects, Inc.

Getting Started

© 2011 CodeObjects, Inc. All rights reserved.

No part of this book may be reproduced, stored in a retrieval system, or transmitted by any means electronic or otherwise without written permission from the publisher.

We have made every effort to make this book as complete and accurate as possible, but no warranty of fitness is implied. The information is provided "as is," and the publisher assumes no liability for any losses or damages arising from the use of this information.

CORADS is a trademark of CodeObjects, Inc. WebSphere is a registered trademark of IBM. Other trademarks are the property of their respective owners.

This manual is confidential and proprietary information provided by CodeObjects, Inc. for clients only.

CodeObjects Inc.
1381 McCarthy Blvd.
Milpitas, CA 95035 USA
<http://www.codeobjects.com>

Table of Contents

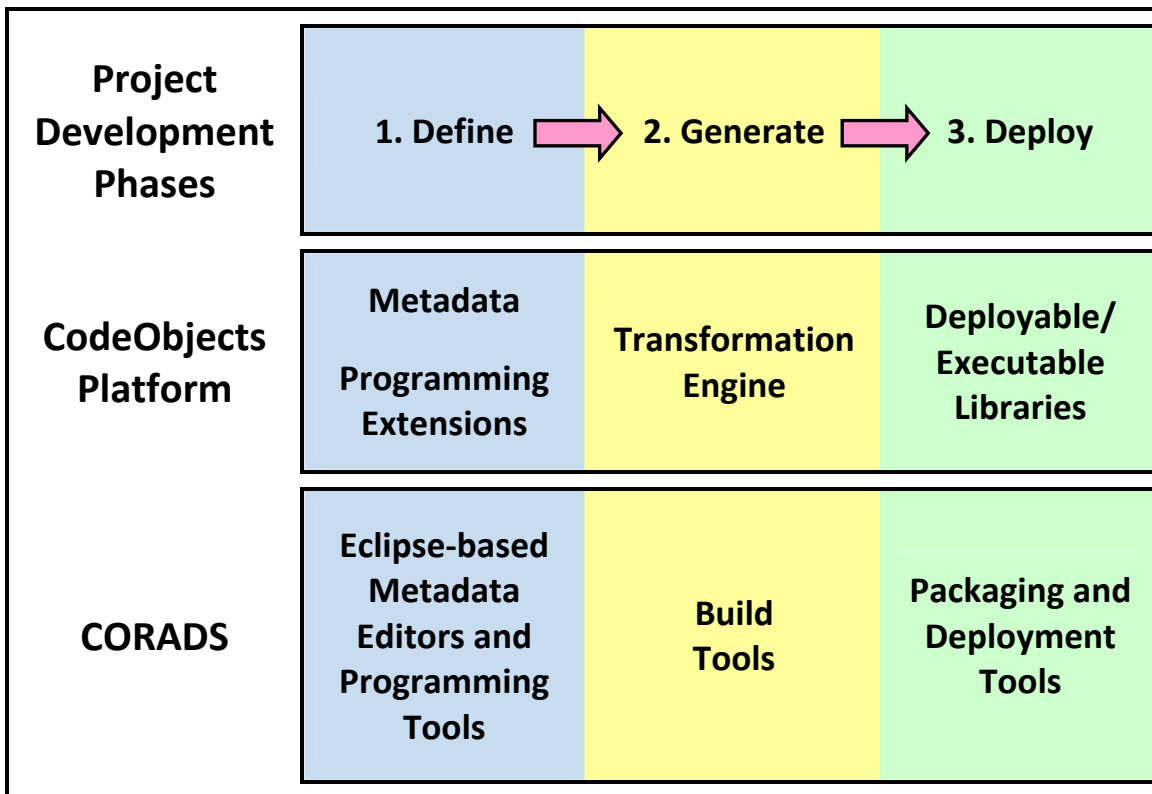
Overview	1
Layers of the CodeObjects Platform Application Model (Metadata).....	2
Deployment.....	3
Customizing Your Application.....	4
CodeObjects Platform Enterprise Components.....	4
System Requirements	5
Development Environment	5
Database Server	6
Testing Environment Application Server	7
Production Application Server	7
Installation	8
Installing the Development Environment	8
Installing Java	8
Installing Eclipse	9
Installing .NET	9
Installing CORADS	10
Unzip the File.....	10
Installation Procedure	11
CodeObjects Platform Enterprise Setup (Optional)	12
Installing the Application Server (Tomcat)	12
Install Tomcat.....	13
Installing the Database Server — SQL Server / Oracle.....	13
Setting up JTA/XA for Microsoft SQL Server.....	14
SQL Server Authentication	15
Enabling XA for MSDTC	15
Create CodeObjects Database Schema.....	16

Production Deployment to WebSphere.....	18
Preliminary Issues.....	18
Setting up WebSphere for CodeObjects Platform	
Applications	18
JDBC Provider	19
Setting Up the WebSphere Shared Library	21
First-Time Deployment	28
Configuring Data Sources	28
Setting Up JMS	35
JNDI Lookup for Service Properties	44
Deploy the Application	45
Updating Your Deployment.....	48

Overview

CodeObjects Platform is a framework and infrastructure system for [Model Driven Development \(MDD\)](#) for enterprise-grade database applications. CodeObjects Platform implements MDD in a powerful, flexible way whereby you specify your application *model* plus any custom *programming extensions*, and the build system generates all of the necessary application artifacts for the desired platform/environment.

The diagram below shows the CodeObjects Platform components that implement the development phases of MDD and the **CodeObjects Rapid Application Development Studio** (CORADS) tools that help you build applications with CodeObjects Platform.



At the heart of CodeObjects Platform is **metadata**. Using the CORADS tools, you define your application with metadata, then generate binaries to deploy to your chosen platform. The project development phases are:

1. **Define the application metadata**

Physical data, and database specifications, business logic, what services are to be provided, what the users can see.

2. **Generate the application**

Transform the metadata into code, merging custom programming extensions.

3. **Deploy the application**

Deploy the applications and run-time components to the application server.

CodeObjects Platform provides a framework for creating applications: specifications for how to create metadata, an engine to create an application from the metadata, and deployable/executable libraries to run the application.

CodeObjects Rapid Application Development Studio (CORADS) is the development environment and tool set for building and maintaining enterprise-grade database-driven applications based on CodeObjects Platform.

Layers of the CodeObjects Platform Application Model (Metadata)

The CodeObjects Platform application model consists of the following specification layers:

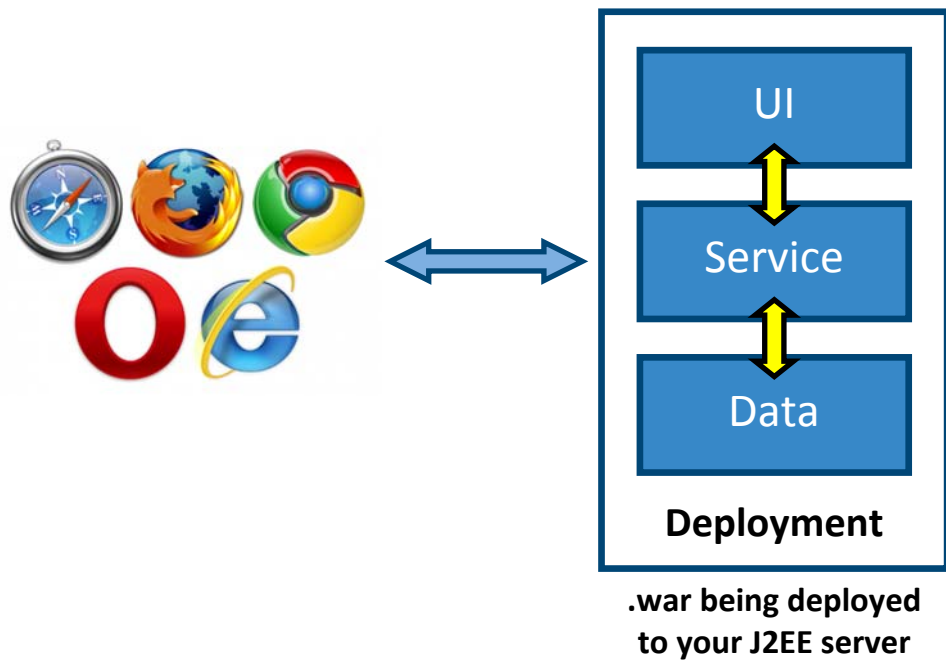
Data Layer	Physical data tables Application data model (data objects, entities, and relationships)
Services Layer / Business Logic	Services to be provided Events and flow
Client Layer / User Interface	Views Widgets (Screen tabs)
Packaging and Deployment	Target client Modules Security: User authentication and granular authorization

For each layer, CORADS provides editors and tools. For example, the tools to specify the UI for a user input screen are very different than the tools that specify the data tables and fields of your database.

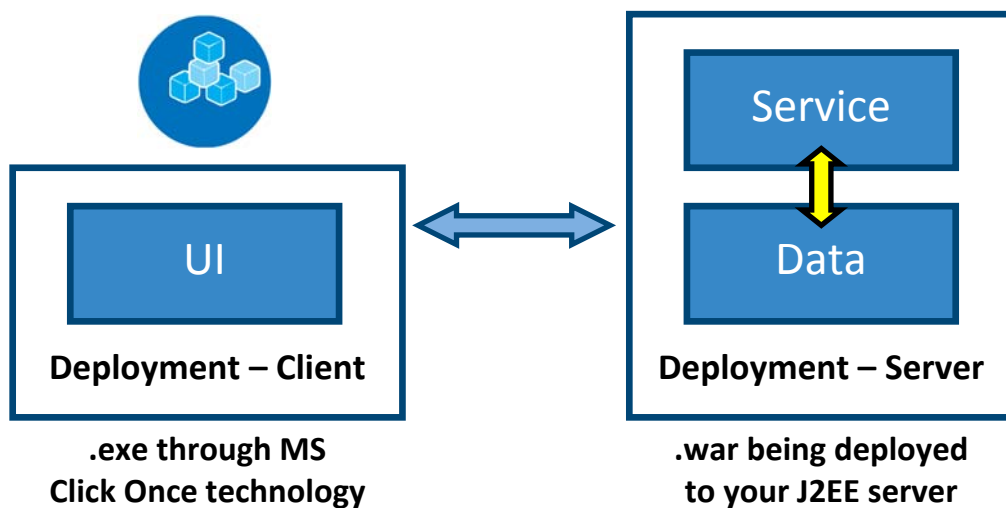
Deployment

CodeObjects Platform supports two deployment models:

1. **HTML based web client:** CORADS creates a J2EE Web Application ARchive (WAR) for deployment.



2. **.NET/Windows based client:** CORADS generates a .NET Windows client executable (.exe) plus a J2EE Web Application ARchive (WAR) for deployment.



Customizing Your Application

In addition to using the CORADS tooling to define application metadata, CodeObjects Platform supports extensive customization including:

- Adding new third-party libraries
- Extending a CORADS module by
 - Creating/modifying a customized HTML layout
 - Creating/modifying custom UI behavior with Java code
- Creating/modifying new modules
- Writing programming extensions to customize modules

Customizations are done through CORADS so that changes are incorporated each time the application is generated.

CodeObjects Platform Enterprise Components

CodeObjects Platform also provides built-in enterprise-grade software components for specialized applications:

- Rating Engine
- Assignment Engine
- Data Quality – Address Cleansing
- Data Audit
- Batch Job – scheduling and execution
- CTI
- Payment Processing (Credit Card and ACH)
- Document Generation (PDF)
- File Attachment
- Party Model
- Data Visibility/Ownership
- Outbound Email Integration
- Work Item
- Common Data Components
 - Notes, Correspondence, Work Items, Employees, and so on.

System Requirements

Installing the CodeObjects Platform / CORADS environment requires that you set up the following:

1. **The CORADS development environment** – the system (computer) that you will use for creating your applications.
2. **A Tomcat-based application server for testing** – may be installed on the same system as the development environment or may be installed on a separate test deployment system.
3. **A database server (Oracle or SQL Server)** – can be located on the development environment system or on a separate server; in a typical installation, the database server is already set up and running, and CORADS simply needs to access it.
4. **A WebSphere application server for production deployment** – usually on a separate system (not part of the development environment system); in a typical installation, the WebSphere server is already set up and running, and CORADS simply needs to access it. This is optional if you are just doing the development.

In this chapter, we discuss the requirements for each of these. The next chapter provides a complete description of how to install system components and build your development and testing environment, as well as the settings that you must adjust to ensure that CORADS applications build and deploy correctly with your database and application servers.

Consult our website at <http://www.codeobjects.com> for the most up-to-date information on specific platform support.

Development Environment

CORADS is based on the Eclipse IDE (<http://www.eclipse.org/>). In this documentation, we're going to assume that you have a basic familiarity with Eclipse. If you haven't used Eclipse, you can get documentation and help at [the Eclipse web site](#).

CORADS generates code in either C# or Java. We assume that you are familiar with your target language in order to create your programming extensions.

CORADS uses the .NET framework. Familiarity is required for custom extensions to desktop deployments, although you won't need to understand .NET to go through most of the basic application development tasks described in the tutorial. The .NET language is only required if you are creating a .NET/Windows client. If you are developing a web client, you need not use the .NET framework.

CORADS is based on J2EE and Service-Oriented Architecture (SOA). You will need some familiarity with the concepts and terminology of J2EE development to work with CORADS.

The development environment requires the following:

- **Operating system** Windows Server 2003, Windows Server 2008, Windows XP, Windows Vista, or Windows 7
- **Memory** 2GB memory minimum, 4GB recommended
- **Disk space**
 - CodeObjects, itself, requires 1 GB of drive space.
 - In addition, you will need space for other components of your development. (This may vary depending on which components you install on your own workstation, and which you install on your network on a separate server.)
 - A complete installation requires at least 3.5 GB and additional drive space for temporary files and application storage.
- **High-performance hardware** If you are working with a large application, with many metadata files, you may experience delay or latency when CORADS is producing the binaries or compilation. In general, we recommend a high performance CPU for the development environment.

Database Server

The installation documentation is based on the following database systems:

- Oracle 10.x
- Microsoft SQL Server 2005, or SQL Server 2008

Other database systems may work but are not supported by CodeObjects, Inc.

Testing Environment Application Server

We recommend a test environment that uses **Apache Tomcat 6.x** (<http://tomcat.apache.org/>) as the application server. We recommend that Tomcat only be used as a development, unit testing, and verification environment, not as your production server.

NOTE

At this time, we support a 32-bit environment only.
Be sure to download the 32-bit version of Tomcat.

Production Application Server

IBM WebSphere 6.x or 7 is the current certified production-grade application server. Refer to the IBM WebSphere website at <http://www-01.ibm.com/software/websphere/#> for more information about the application server. For information about deployment, see “Production Deployment to WebSphere” on page 18.

Installation

Installing the Development Environment

Installing Java

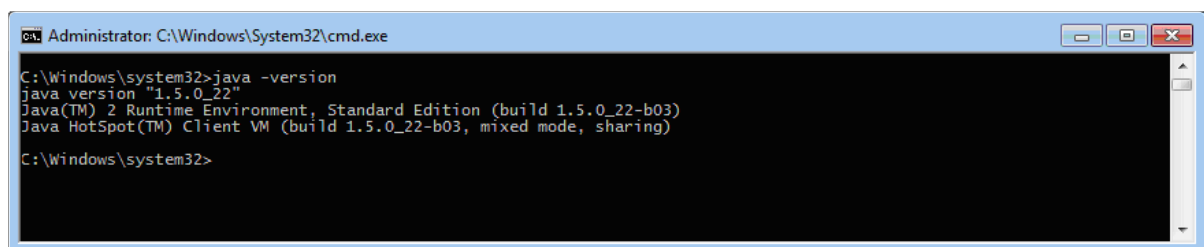
1. Install the Java SE Development Kit 6 (32-bit). You can download the Java Developer Kit (JDK) at:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

2. Set the environment variable **JAVA_HOME** to the root directory of the SDK installation. Based on the location where you installed Java and your specific version, you might set, for example:

SET JAVA_HOME=C:\Program Files\Java\jdk1.6.0_26

3. Verify that Java has been installed correctly. To ensure that Java is on the path, open a Command window and enter **java -version** to show the effective java version. Verify that the reported version is the one that you installed.



```
Administrator: C:\Windows\System32\cmd.exe
C:\Windows\system32>java -version
java version "1.5.0_22"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_22-b03)
Java HotSpot(TM) Client VM (build 1.5.0_22-b03, mixed mode, sharing)
C:\Windows\system32>
```

Only if your system has an unusual existing installation, you might append the java \bin subdirectory location to your system path as a troubleshooting measure. For example, to append C:\Program Files\Java\jdk1.6.0_26\bin to the existing path, enter:

PATH %PATH%;C:\Program Files\Java\jdk1.6.0_26\bin

Installation

Installing Eclipse

1. Download Eclipse 3.6.x, the Eclipse IDE for Java Developers (32-bit) at:
<http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/heliossr2>
2. Unzip the downloaded file to the directory where you are installing Eclipse. In this document, we refer to this Eclipse directory as:
<EclipseInstallRoot>
3. Create a shortcut to Eclipse with the following suggested parameters appended to the target:
-vmargs -Xms256m -Xmx784m -XX:PermSize=64m -XX:MaxPermSize=128m

For more information about Eclipse parameters and performance tuning, consult the Eclipse documentation at:

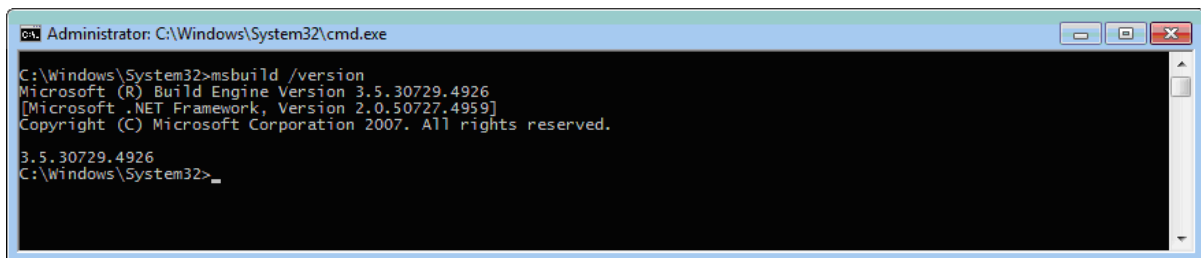
<http://eclipse.org>

Installing .NET

1. Install .NET Framework 3.5:
<http://www.microsoft.com/downloads/en/details.aspx?FamilyID=333325fd-ae52-4e35-b531-508d977d32a6&DisplayLang=en>

Be sure to install the security updates, which may require a separate step.

2. Verify that the .Net 3.5 version has been installed correctly. Open a Command window and enter **msbuild /version** which should show 3.5.x as appropriate.



```
Administrator: C:\Windows\System32\cmd.exe
C:\Windows\System32>msbuild /version
Microsoft (R) Build Engine Version 3.5.30729.4926
[Microsoft .NET Framework, Version 2.0.50727.4959]
Copyright (C) Microsoft Corporation 2007. All rights reserved.

3.5.30729.4926
C:\Windows\System32>
```

NOTE

If you are planning to develop applications that will be deployed to users as a Windows desktop application, you may want to install a C# IDE (Integrated Development Environment) such as Microsoft Visual Studio.

Installing CORADS

The CodeObjects Platform and CORADS are shipped in a single zip file. Contact CodeObjects if you did not receive such a zip file for CORADS.

Unzip the File

Unzip the file *codeobjects.zip* to the directory where you are installing the CodeObjects Platform. In this document, we refer to this CodeObjects directory as:

<CodeObjectsInstallRoot>

After you unzip the file, the <CodeObjectsInstallRoot> directory contains the following:

modules	The CodeObjects Platform, runtime libraries, and applications configured for customers.
third-party	Non-CodeObjects Platform libraries, required for the CodeObjects Platform application.
libraries	CodeObjects Platform runtime libraries. These are provided as binaries and cannot be changed by the customer.
application	CodeObjects Platform base application modules, which can be augmented / extended by the customer.
extensions	Customer modules initially developed by CodeObjects.
resources	Language files used in the CodeObjects Platform build framework files.
CORADS	Local Eclipse update site for CORADS.
ProductEnterprise	The CodeObjects Platform rating and underwriting authoring program.
database	The database SQL scripts to create the table schemas for MS SQL and Oracle databases.
component-model	The database SQL script to create the table schemas used by CodeObjects common components.
pac-model	The database SQL scripts to create the table schema used by CodeObjects Property & Causality applications.
finished-tutorial-model	The database SQL scripts to create the table schema used by the tutorial application.

Installation Procedure

1. Run Eclipse using the previously created shortcut. If prompted, choose a workspace location. (Your Eclipse workspace folder contains temporary files. The workspace can be located wherever you want.)
2. Install the Database Development Tools:
 - a. From the Eclipse menus, click **Help** → **Install New Software**.
 - b. From the **Work With** list, choose **Helios** – <http://download.eclipse.org/releases/helios>
Note that this step can take several few minutes because it requires files to be downloaded from the Internet. Wait for the **Pending** indicator to display a list of items before proceeding to the next step.
 - c. Select the **Group items by Category** check box if it not already selected.
 - d. From the list of available Eclipse features, choose **Database Development**.
 - e. Click the **Finish** button, accept any license terms that appear, and restart Eclipse when prompted.
3. Wait for Eclipse to restart, and select the workspace you used previously if prompted.
4. Install the CORADS software.
 - a. From the menu, click **Help** → **Install New Software**.
 - b. Click the **Add** button.
 - c. In the **Add Repository** box, set **Name** to **CodeObjects**.
 - d. Click the **Local** button and choose **<CodeObjectsInstallRoot>/CORADS** in the file browser. Although CORADS is downloaded through a local update site, Internet access is required for this step so that other prerequisites can be resolved.
 - e. Clear the **Group items by category** check box if it is selected.
 - f. Select the check box next to **CodeObjects Rapid Application Development Studio**, and then click the **Next** button.
 - g. Again, click the **Next** button to accept the defaults.
 - h. Accept the license terms and click the **Finish** button.
 - i. Click **OK** to accept installation of unsigned content.
 - j. Click the **Restart Now** button to restart Eclipse.

CodeObjects Platform Enterprise Setup (Optional)

This step installs the Rules Editors used for the rating and underwriting feature.

1. From the Eclipse menu, click **Help** → **Install New Software**.
2. In the dialog box, click the **Add** button.
3. In the **Add Repository** box, set **Name** to **Apache Drools 3**, and set **Location** to **<http://downloads.jboss.com/drools/updatesite3.5/>**
4. Click **OK**.
5. Ensure that the **Group Items By Category** check box is selected.
6. Select the root level **JBoss Drools** check box.
7. Click the **Finish** button, accept any license terms that appear, and restart Eclipse when prompted.

Installing the Application Server (Tomcat)

We recommend Apache Tomcat as your test environment. We currently support:
Tomcat 6.x (32-bit)

In this manual, we assume that you will test with Tomcat. For instructions about deploying to WebSphere (when you deploy your application for production), see “Production Deployment to WebSphere” starting on page 18.

You can download Tomcat at <http://tomcat.apache.org/download-60.cgi>. Any of the core versions will work, as long as it's a 32-bit version, but maintain any version compatibility required throughout your organization.

NOTE

Do not use the 64-bit version, even if you have a 64-bit server.

Install Tomcat

To install the file:

1. Unzip the downloaded file to the directory where you are installing Tomcat. In this document, we refer to this installation directory as **<TomcatInstallRoot>**
2. Set the **CATALINA_OPTS** environment variable to ensure that you have allocated enough memory for Tomcat. The following example is a suggestion. The actual value you should use depends on the size of your application:

-Xmx750m -XX:MaxPermSize=400m

For more information about Tomcat parameters, consult the Tomcat documentation.

3. Copy all the files at
<CodeObjectsInstallRoot>\modules\resources\env\JTA\Atomikos
to
<TomcatInstallRoot>\lib

Installing the Database Server — SQL Server / Oracle

You can install the database server on your own workstation or on another (possibly dedicated) database server computer in your network. It is beyond the scope of this document to describe how to install the SQL Server or Oracle database software. Consult your Microsoft or Oracle documentation for that information. In this document, we assume that you have an installed database server.

CORADS requires the following additional installation in order to work with your database:

- [Java Transaction Architecture \(JTA\)](#) must be installed/enabled for Microsoft SQL Server.

In addition, to work through the CORADS tutorials and the PolicyEnterprise and BillingEnterprise applications, you will need:

- CORADS database schemas.
- CORADS seed data populated.

The steps to install these components on your database server are described in detail below.

Setting up JTA/XA for Microsoft SQL Server

Follow these steps to set up Java Transaction Architecture (JTA) and transaction processing (XA) for CORADS database access (adapted from the Microsoft documentation).

Search for “JDBC Driver” at <http://microsoft.com/downloads>. Download a copy of Microsoft SQL Server JDBC Driver 3.0 or above.

1. Extract the driver.
2. Create the folder

**c:\Program Files\Microsoft SQL Server JDBC
Driver\sqljdbc_3.0\enu**

3. Copy sqljdbc4.jar into the folder. (This is for Java 6. For Java 5, copy sqljdbc.jar instead.)
4. Append the following to the CLASSPATH:

**;C:\Program Files\Microsoft SQL Server JDBC
Driver\sqljdbc_3.0\enu\sqljdbc4.jar**

5. Copy **sqljdbc_xa.dll** from the JDBC installation directory to the Binn directory of every SQL Server computer that will participate in distributed transactions.
 - a. If you are using XA transactions with a 32-bit SQL Server, put the **sqljdbc_xa.dll** file in the x86 folder, even if the SQL Server is installed on a x64 processor.
 - b. If you are using XA transactions with a 64-bit SQL Server on an x64 processor, put the **sqljdbc_xa.dll** file in the x64 folder.
 - c. If you are using XA transactions with a 64-bit SQL Server on an Itanium processor, put the **sqljdbc_xa.dll** file in the IA64 folder.

For example, you might put the dll file into
C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Binn
There may be multiple binn folders.

6. Execute the database **script xa_install.sql** on every SQL Server instance that will participate in distributed transactions.

This script installs the extended stored procedures that are called by **sqljdbc_xa.dll**. These extended stored procedures implement distributed transaction and XA support for the Microsoft SQL Server JDBC Driver. You must run this script as an administrator of the SQL Server instance.

SQL Server Authentication

MS SQL Server comes with two authentication models, SQL Server Authentication and Windows Authentication. CORADS currently certifies SQL Server Authentication only. If your local SQL Server is currently using Windows Authentication, configure your SQL Server to accept SQL Server Authentication.

Refer to Microsoft SQL Server documentation to configure SQL Server Authentication.

Enabling XA for MSDTC

The driver documentation includes detailed instructions for configuring XA data sources with the Microsoft Distributed Transaction Coordinator (MS DTC). For convenience, follow the steps below to configure XA for CORADS [adapted from the Microsoft documentation].

To Configure XA Data Sources for Windows 7 or Windows Vista

1. Click the **Start** button.
 - In the **Start Search** box, type **dcomcnfg** and then press ENTER to open **Component Services**.
 - or–
 - In the **Start Search** box, type **%windir%\system32\comexp.msc** and then press ENTER to open **Component Services**.
2. Expand **Component Services**, **Computers**, **My Computer**, and **Distributed Transaction Coordinator**.
3. Right-click **Local DTC** and choose **Properties**.
4. The **Local DTC Properties** dialog box opens. Click the **Security** tab.
5. Select the **Enable XA Transactions** check box, and then click **OK**. This restarts the **MS DTC** service.
6. Click **OK** again to close the **Properties** dialog box, and then close **Component Services**.
7. Stop and then restart **SQL Server** to ensure that it syncs up with the **MS DTC** changes.

To Configure XA Data Sources for Windows Server 2003 or Windows XP

1. From **Control Panel**, open **Administrative Tools**, and then open **Component Services**. You can also click the **Start** button, click **Run**, type **dcomcnfg** in the **Open** box, and then press **OK** to open **Component Services**.

2. Expand Component Services, Computers and right-click **My Computer**, and then select **Properties**.
3. Click the **MSDTC** tab, and then click **Security Configuration**.
4. Select the **Enable XA Transactions** check box, and then click **OK**. This will cause a MS DTC service restart.
5. Click **OK** again to close the **Properties** dialog box, and then close **Component Services**.
6. Stop and then restart SQL Server to ensure that it syncs up with the **MS DTC** changes. The Distributed Transaction Coordinator service might or might not be running. If it is not running, at a command prompt:
 - a. Enter **msdtc -resetlog**
 - b. Enter **net start msdtc**
 - c. Configure the computer to start the service at startup.

Create CodeObjects Database Schema

This section explains how to create the db schemas for CodeObjects.

CodeObjects provides these database tables for the Microsoft SQL Server and Oracle. You may create other database tables for your applications.

1. Table schemas for Common Components, such as tables for notes, work items, correspondences, employees, and so on.
2. Table schemas for Property & Casualty applications, such as tables for insurance products, coverages, policies, payments, and so on.
These tables depend on the tables in common components.

Additionally, there are table schemas for the CORADS tutorial application, which also depend on the tables in common components.

The following information describes the *database* folder contents:

database The database SQL scripts to create the table schemas for MS SQL and Oracle databases.

component-model The database SQL script to create the table schemas used by CodeObjects common components.

pac-model The database SQL scripts to create the table schema used by CodeObjects Property & Causality applications.

When running the SQL scripts here, be sure to run the SQL scripts in *component-model* first.

finished-tutorial-model The database SQL scripts to create the table schema used by the tutorial application.

When running the SQL scripts here, be sure to run the SQL scripts in *component-model* first.

The application scope	Steps to create the table schemas
To build your own application on top of CodeObjects common components	1. Create your own database. 2. Run the SQL scripts under <i>component-model</i> against the database you created.
To go through the tutorial application	1. Create a new database with the name <i>CORADS_TUTORIAL</i>. 2. Run the SQL scripts under <i>component-model</i> against the <i>CORADS_TUTORIAL</i>. database. 3. Run the SQL scripts under <i>finished-tutorial-model</i> against the <i>CORADS_TUTORIAL</i>. database.
To build your own application on top of CodeObjects Property and Casualty application.	1. Create your own database. 2. Run the SQL scripts under <i>component-model</i> against the database you created. 3. Run the SQL scripts under <i>pac-model</i> against the database you created.

Production Deployment to WebSphere

Preliminary Issues

When deploying to WebSphere, you must:

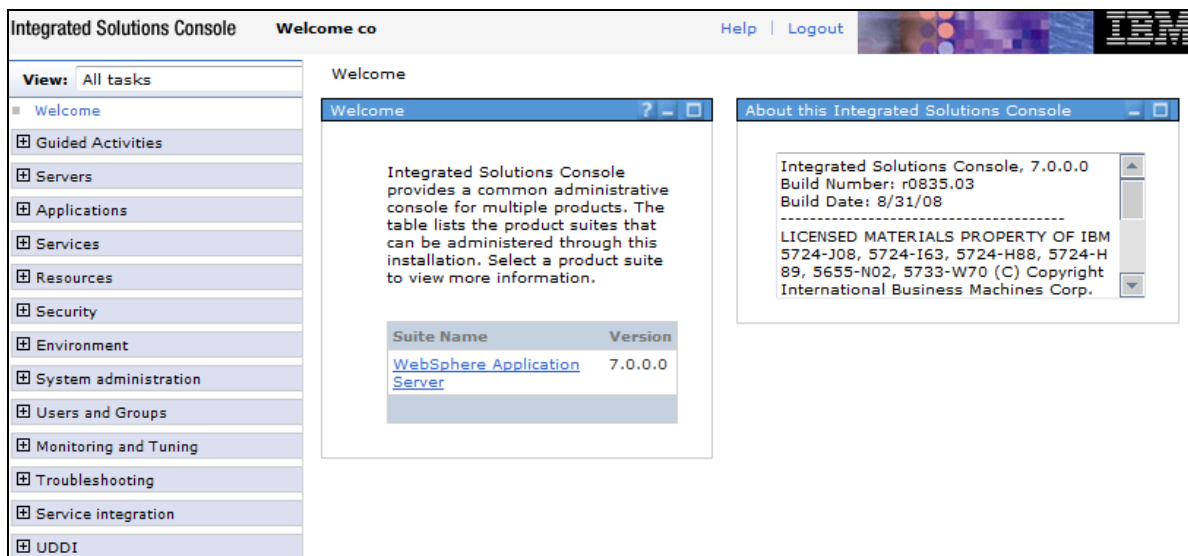
- **Make some initial configuration changes** to allow CodeObjects Platform applications to run.
- **The first time you deploy**, you must configure application-specific settings and then transfer your WAR file to the WebSphere server.
- **For subsequent deployments**, you must check for specific application changes and then simply re-deploy the WAR file.

This is described in detail below.

Setting up WebSphere for CodeObjects Platform Applications

In this section, we describe configuration settings that allow CodeObjects Platform applications to run with your WebSphere Application Server (WAS) installation. We assume that WebSphere be installed and running correctly.

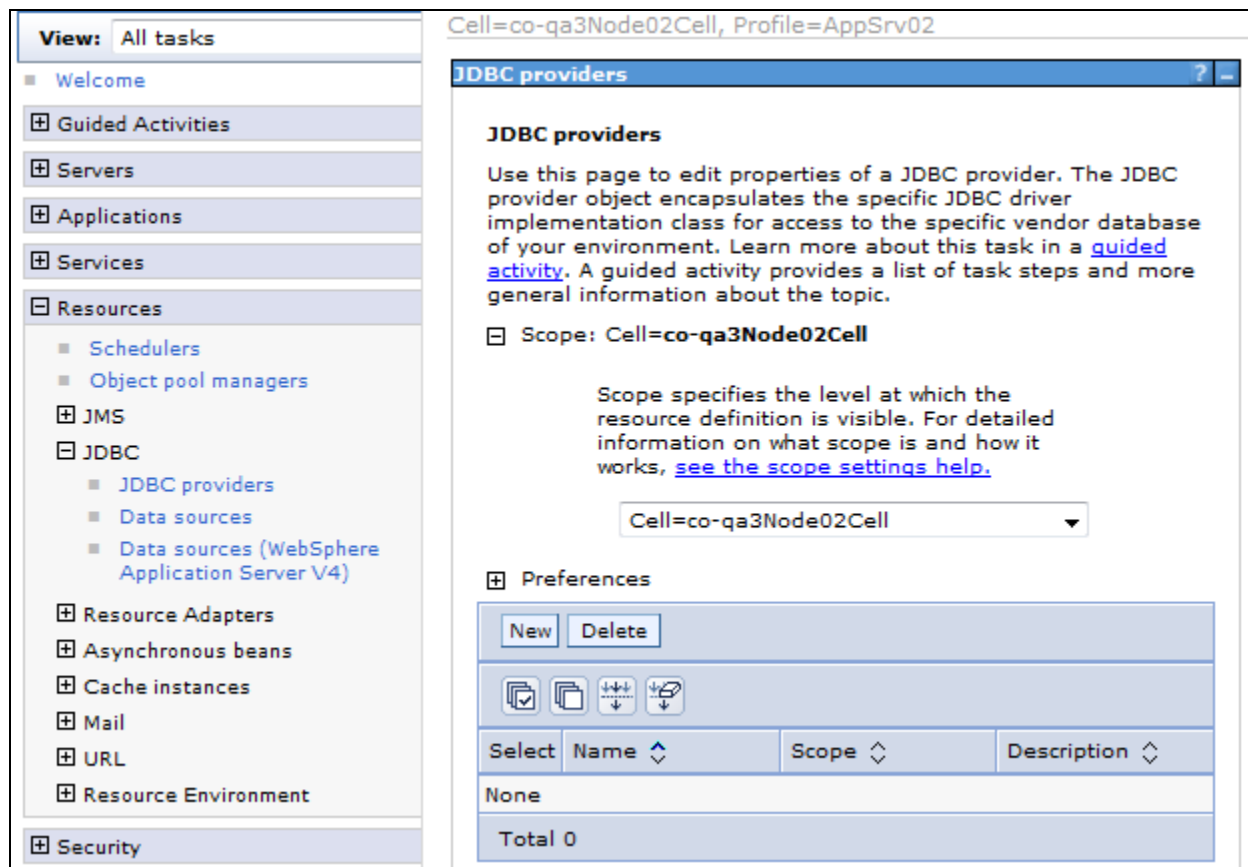
To configure the settings below, you must be logged in to the WebSphere Integrated Solutions Console in your browser.



JDBC Provider

JDBC Providers encapsulate specific JDBC drivers for your database environment. Set up the JDBC provider once, at installation, from the WebSphere Integrated Solutions Console:

1. In the Tasks pane, expand **Resources** → **JDBC** and click **JDBC providers**. Select the scope in which your application will run.



2. In the Configuration pane, click **New**.

3. Set the **Database type**, to **Oracle**.

Create a new JDBC Provider

Create a new JDBC Provider

→ **Step 1: Create new JDBC provider**

Step 2: Enter database class path information

Step 3: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope
cells:co-qa3Node02Cell

* Database type
Oracle

* Provider type
Oracle JDBC Driver

* Implementation type
XA data source

* Name
Oracle JDBC Driver (XA)

Description
Oracle JDBC Driver (XA)

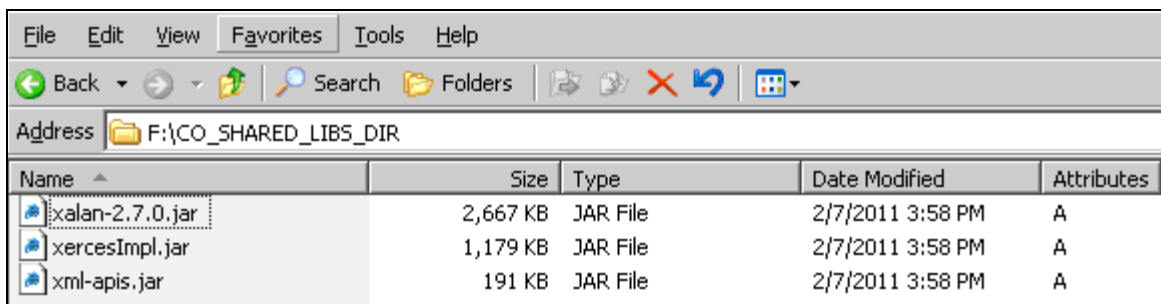
Next Cancel

4. Set the **Provider type**, to **Oracle JDBC Driver**.
5. Set the **Implementation type**, to **XA data source**.
6. Click **Next**.
7. Specify the directory location for **ojdbc6.jar**, which is saved as WebSphere variable **\${ORACLE_JDBC_DRIVER_PATH}**. This is typically something similar to **\${WASInstallationRoot}/AppServer/lib** where the WAS installation root could be **/IBM/WebSphere**
8. Click **Next**.
9. When you are satisfied with the summary of the new JDBC provider you are about to create, click **Finish** to create it.

Setting Up the WebSphere Shared Library

CodeObjects Platform provides XML processing libraries for use with WebSphere. You must copy these libraries to a folder on the WebSphere server. Setting up the shared libraries prevents class loader issues at deployment. To set up the libraries:

1. Create a local directory somewhere on the computer where WebSphere is installed.
This directory is referred to as **<CO_SHARED_LIBS_DIR>**
2. Place the following *.jars* in your new folder:
`<CodeObjectsInstallRoot>\modules\third-party\xalan\java\dist\ xalan-2.7.0.jar`
`<CodeObjectsInstallRoot>\modules\third-party\xml-apis\java\dist\ xml-apis.jar`
`<CodeObjectsInstallRoot>\modules\third-party\xerces-impl\java\dist\ xercesImpl.jar`



Name	Size	Type	Date Modified	Attributes
xalan-2.7.0.jar	2,667 KB	JAR File	2/7/2011 3:58 PM	A
xercesImpl.jar	1,179 KB	JAR File	2/7/2011 3:58 PM	A
xml-apis.jar	191 KB	JAR File	2/7/2011 3:58 PM	A

3. Log in to the **Integrated Solutions Console**, and in the Tasks pane, expand **Environment** and click **Shared libraries**.



4. In the Configuration pane, set the **Scope** to **Cell** and click the **New** button to create a new shared library.

The screenshot shows the 'Shared Libraries' configuration page. At the top, there's a title bar 'Shared Libraries' with a help icon. Below it, the section 'Shared Libraries' is followed by a description: 'Use this page to define a container-wide shared library that can be used by deployed applications.' A checkbox labeled 'Scope: Cell=co-qa2Node01Cell' is checked. Below this, a text box explains: 'Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help.](#)' A dropdown menu is shown with 'Cell=co-qa2Node01Cell' selected and highlighted by a red rectangle. Below the dropdown is a 'Preferences' section with a '+' icon. It contains two buttons: 'New' (highlighted with a red rectangle) and 'Delete'. Below these are four icons: a folder, a document, a list, and a search icon. A table with columns 'Select', 'Name', and 'Description' is shown. The table is currently empty, with 'None' in the 'Select' column and 'Total 0' at the bottom.

Shared Libraries

Use this page to define a container-wide shared library that can be used by deployed applications.

☒ Scope: Cell=co-qa2Node01Cell

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help.](#)

Cell=co-qa2Node01Cell

Preferences

New Delete

Select Name Description

None

Total 0

5. Set the **Name** to **CO_SHARED_LIBS**, and set the **Classpath** to the following (.jars must be delimited with new lines):

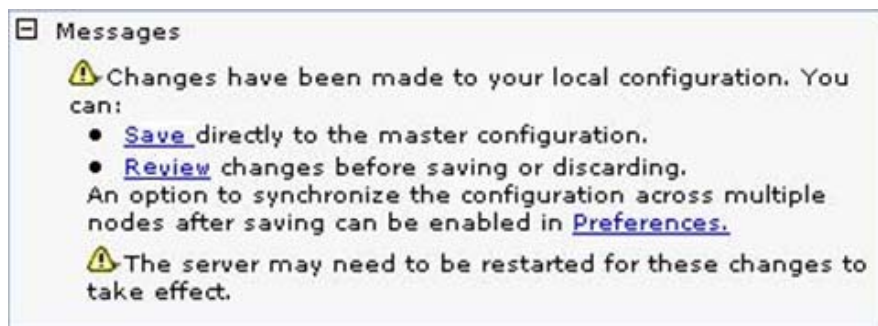
<CO_SHARED_LIBS_DIR>/xalan-2.7.0.jar

<CO_SHARED_LIBS_DIR>/xml-apis.jar

<CO_SHARED_LIBS_DIR>/xercesImpl.jar

The screenshot shows the 'Configuration' tab in the WebSphere Configuration Console. Under 'General Properties', the 'Name' field is set to 'CO_SHARED_LIBS' and the 'Classpath' field contains the following paths, each on a new line: 'F:\CO_SHARED_LIBS_DIR\xalan-2.7.0.jar', 'F:\CO_SHARED_LIBS_DIR\xercesImpl.jar', and 'F:\CO_SHARED_LIBS_DIR\xml-apis.jar'. The 'Scope' is set to 'cells:co-qa2Node01Cell'. The 'Class Loading' section has the checkbox 'Use an isolated class loader for this shared library' unchecked. At the bottom are buttons for 'Apply', 'OK', 'Reset', and 'Cancel'.

6. Click **OK** and save changes when prompted.



7. In the Tasks pane, navigate to **Servers**→**Server Types** and click **WebSphere application servers**. In the Configuration pane, drill down on the application server that will be hosting the CodeObjects Platform application.

The screenshot shows the Integrated Solutions Console interface. On the left, the 'Servers' category is expanded, and 'Server Types' is selected, with 'WebSphere application servers' highlighted. On the right, the 'Application servers' page is displayed, showing a table of application servers. The table has columns for 'Name' and 'Node'. The first entry is 'server1' under the 'co-qa2Node02' node. The total count is 1.

Integrated Solutions Console Welcome

View: All tasks

- Welcome
- Guided Activities
- Servers
 - Server Types**
 - WebSphere application servers
 - WebSphere MQ servers
 - Web servers
- Applications
- Services
- Resources
- Security
- Environment
 - Virtual hosts
 - Update global Web server plug-in configuration
 - WebSphere variables
 - Shared libraries
 - Replication domains
- Naming
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=co-qa2Node01Cell, Profile=AppSrv01

Application servers

Use this page to view a list of the application servers. Use this page to change the status of a specific application server.

Preferences

Name	Node
You can administer the following resources:	
server1	co-qa2Node02
Total 1	

8. Drill down on **Java and Process Management** → **Class loader**.

The image shows the 'General Properties' configuration page for an application server named 'server1'. The 'Node name' is 'co-qa2Node02'. Under 'Access to internal server classes', 'Allow' is selected. In the 'Server-specific Application Settings' section, 'Classloader policy' is 'Single' and 'Class loading mode' is 'Classes loaded with parent class loader first'. On the right, the 'Container Settings' and 'Applications' sections are visible. Under 'Server Infrastructure', 'Java and Process Management' is expanded, and 'Class loader' is highlighted with a red box.

General Properties

Name: server1

Node name: co-qa2Node02

☐ Run in development mode

☒ Parallel start

☐ Start components as needed

Access to internal server classes: Allow

Server-specific Application Settings

Classloader policy: Single

Class loading mode: Classes loaded with parent class loader first

Buttons: Apply, OK, Reset, Cancel

Container Settings

- Session management
- SIP Container Settings
- Web Container Settings
- Portlet Container Settings
- EJB Container Settings
- Container Services
- Business Process Services

Applications

- Installed applications

Server messaging

- Messaging engines
- Messaging engine inbound transports
- WebSphere MQ link inbound transports
- SIB service

Server Infrastructure

- Java and Process Management
 - Class loader**

9. Click the **New** button to create a new Class loader.

The image shows the 'Class loader' configuration page for 'server1'. It includes a breadcrumb trail 'Application servers > server1 > Class loader'. Below the instructions, there are 'New' and 'Delete' buttons. The 'New' button is highlighted with a red box. Below the buttons are icons for creating, cloning, and deleting class loaders. A table with columns 'Class loader ID' and 'Class loader order' is shown, currently containing no entries. The status 'Total 0' is displayed at the bottom.

Application servers > server1 > Class loader

Use this page to configure class loaders.

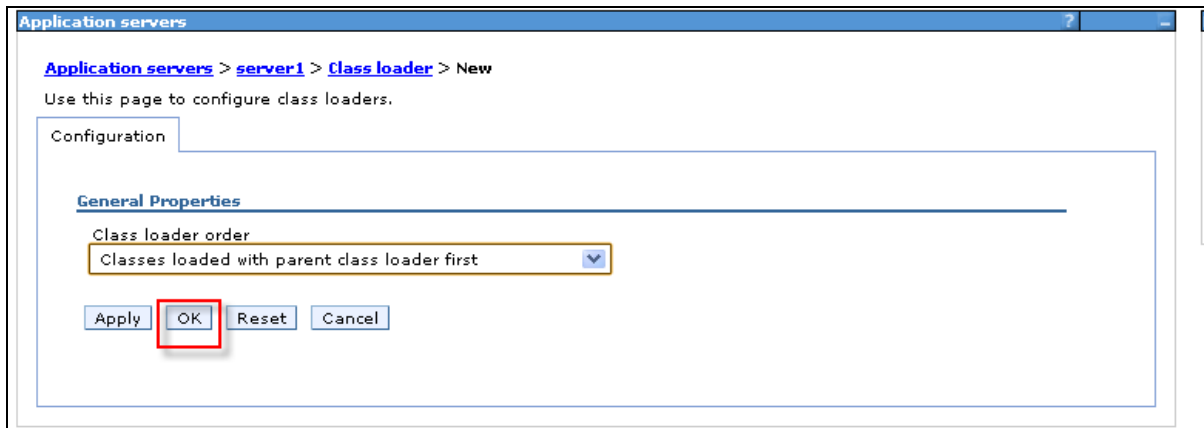
Preferences

New Delete

Icons: Create, Clone, Delete

Select	Class loader ID	Class loader order
None		
Total 0		

10. Set the **Class loader order** to **Classes loaded with parent class loader first**.
Click **OK** and save changes.



Application servers

[Application servers](#) > [server1](#) > [Class loader](#) > New

Use this page to configure class loaders.

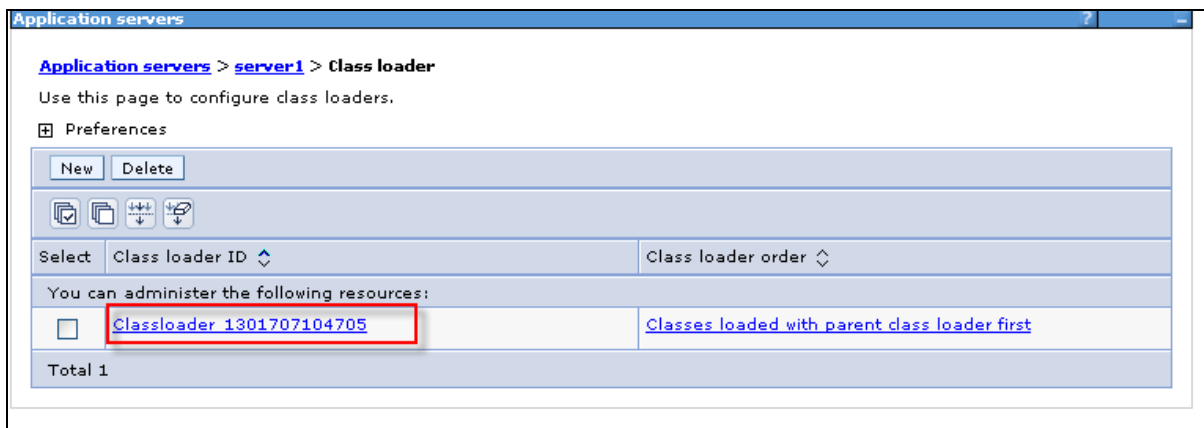
Configuration

General Properties

Class loader order
Classes loaded with parent class loader first

Apply OK Reset Cancel

11. Drill down into the new **Class loader**.



Application servers

[Application servers](#) > [server1](#) > [Class loader](#)

Use this page to configure class loaders.

Preferences

New Delete

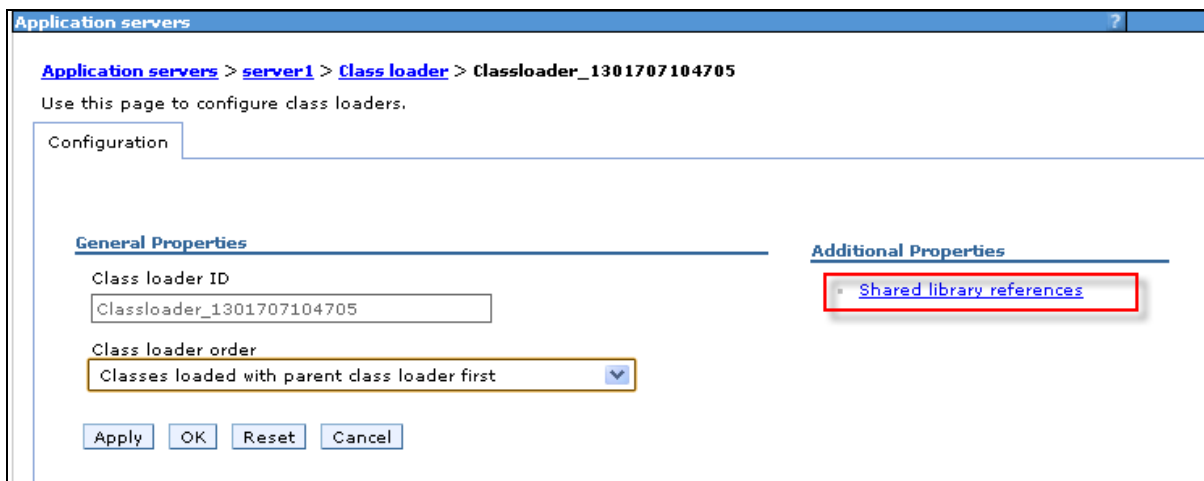
Select Class loader ID Class loader order

You can administer the following resources:

☐	Classloader_1301707104705	Classes loaded with parent class loader first
--------------------------	---	---

Total 1

12. Drill down into **Shared library references**.



Application servers

[Application servers](#) > [server1](#) > [Class loader](#) > [Classloader_1301707104705](#)

Use this page to configure class loaders.

Configuration

General Properties

Class loader ID
Classloader_1301707104705

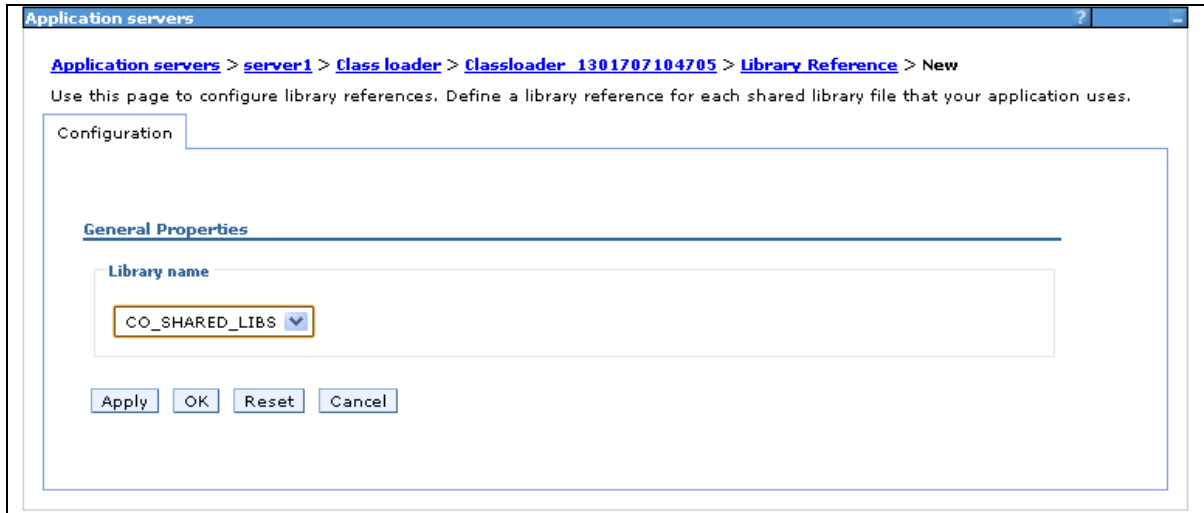
Class loader order
Classes loaded with parent class loader first

Apply OK Reset Cancel

Additional Properties

Shared library references

13. Click the **Add** button and under **Library name**, add **CO_SHARED_LIBS** to the class loader. Click **OK** and save changes.



14. Restart the profile to complete reloading of the application server.

This completes the setup.

First-Time Deployment

This section shows how to deploy applications to IBM WebSphere v7 Application Server. Before deploying, WebSphere must be installed and configured.

To configure the following settings, you must be logged in to the WebSphere Integrated Solutions Console in your browser.

The first time you deploy your application, you must:

1. Configure access to your database (data source).
2. Set up JMS, including setting up topics and topic connections as required by your application.
3. Set up JNDI for your application's system properties.
4. Deploy the WAR file.

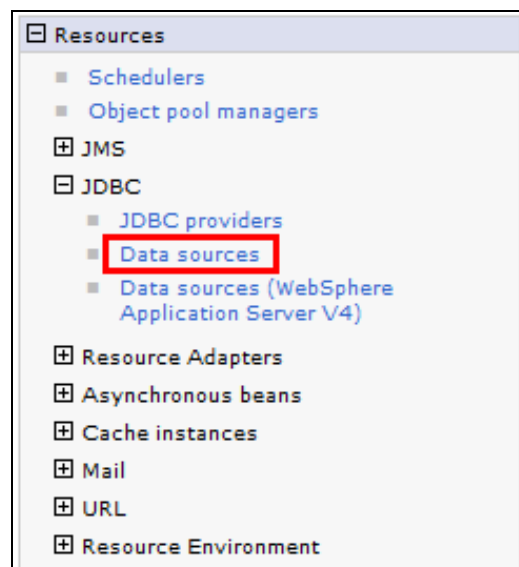
These following steps describe the process in detail.

Configuring Data Sources

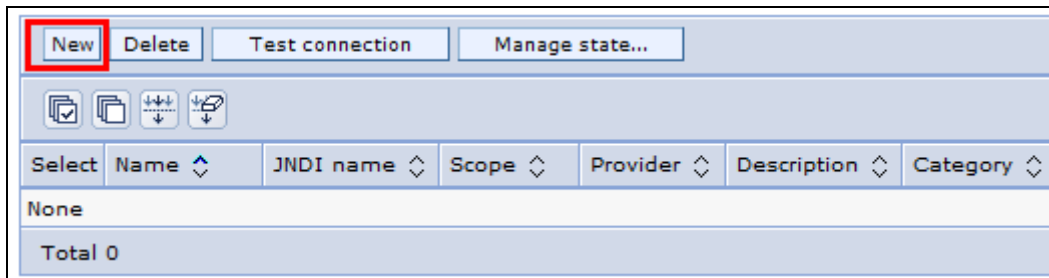
Each CodeObjects Platform application defines a JNDI name for Data Source lookup. The JNDI name is specified in the application's *.application* metadata file. In order for the application to be able to connect to a database, you must create a Data Source with the given JNDI name.

Set up the data source settings for each database that you will be using, from the WebSphere Integrated Solutions Console:

1. In the Tasks pane, expand **Resources** → **JDBC** and click **Data sources**.



- In the Configuration pane, click **New**.



- Specify a unique **Data source name**.

The screenshot shows a wizard titled 'Enter basic data source information'. On the left is a sidebar with five steps: 'Step 1: Enter basic data source information' (selected), 'Step 2: Select JDBC provider', 'Step 3: Enter database specific properties for the data source', 'Step 4: Setup security aliases', and 'Step 5: Summary'. The main area contains instructions: 'Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.' It also includes a requirement: 'Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.' Below this are input fields for 'Scope' (containing 'cells:co-qa3Node02Cell'), '* Data source name' (containing 'MyDatasourceName'), and '* JNDI name' (containing 'jdbc/MyDatasource'). At the bottom are 'Next' and 'Cancel' buttons.

- Specify the **JNDI name**, for example **jdbc/MyDatasource**. This is a string defined in the application's metadata. Click **Next**.

- Click the option button **Select the existing JDBC provider**, and choose **Oracle JDBC Driver (XA)**.

Step 1: Enter basic data source information

→ **Step 2: Select JDBC provider**

Step 3: Enter database specific properties for the data source

Step 4: Setup security aliases

Step 5: Summary

Select JDBC provider

Specify a JDBC provider to support the datasource. If you choose to create a new JDBC provider, it will be created at the same scope as the datasource. If you are selecting an existing JDBC provider, only those providers at the current scope are available from the list.

☐ Create new JDBC provider

☒ Select an existing JDBC provider

Oracle JDBC Driver (XA) ▼

Previous Next Cancel

- Click **Next**.
- Specify the **URL** to the Oracle database.

Step 1: Enter basic data source information

Step 2: Select JDBC provider

→ **Step 3: Enter database specific properties for the data source**

Step 4: Setup security aliases

Step 5: Summary

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

Name	Value
* URL	jdbc:oracle:thin:@co-dev3:15;

* Data store helper class name
Oracle11g data store helper ▼

☒ Use this data source in container managed persistence (CMP)

Previous Next Cancel

- Set the **Data store helper class name** to **Oracle11g data store helper**.
- Click **Next**.

10. Leave all selections for the **Setup security aliases** step alone.

The screenshot shows a wizard interface with a left sidebar and a main content area. The sidebar lists five steps: Step 1: Enter basic data source information, Step 2: Select JDBC provider, Step 3: Enter database specific properties for the data source, Step 4: Setup security aliases (highlighted with a yellow arrow), and Step 5: Summary. The main content area is titled 'Setup security aliases' and contains the instruction 'Select the authentication values for this resource.' Below this are four dropdown menus, all set to '(none)': 'Authentication alias for XA recovery', 'Component-managed authentication alias', 'Mapping-configuration alias', and 'Container-managed authentication alias'. A note at the bottom states: 'Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.' Below the note are two blue underlined links: 'Global J2C authentication alias' and 'Security domains'. At the bottom of the wizard are three buttons: 'Previous', 'Next', and 'Cancel'.

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

→ Step 4: Setup security aliases

Step 5: Summary

Setup security aliases

Select the authentication values for this resource.

Authentication alias for XA recovery
(none) ▼

Component-managed authentication alias
(none) ▼

Mapping-configuration alias
(none) ▼

Container-managed authentication alias
(none) ▼

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

[Global J2C authentication alias](#)
[Security domains](#)

Previous Next Cancel

11. Click **Next**.

12. When the **Summary** of the new data source you are about to create is acceptable, click **Finish** to create it. Save the configuration.

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

Step 4: Setup security aliases

→ **Step 5: Summary**

Summary

Summary of actions:

Options	Values
Scope	cells:co-qa3Node02Cell
Data source name	MyDatasourceName
JNDI name	jdbc/MyDatasouce
Select an existing JDBC provider	Oracle JDBC Driver (XA)
Implementation class name	oracle.jdbc.xa.client.OracleXADataSource
URL	jdbc:oracle:thin:@co-dev3:1521/PNC10
Data store helper class name	com.ibm.websphere.rsadapter.Oracle11gDataStoreHelper
Use this data source in container managed persistence (CMP)	true
Authentication alias for XA recovery	(none)
Component-managed authentication alias	(none)
Mapping-configuration alias	(none)
Container-managed authentication alias	(none)

Previous Finish Cancel

13. Drill down on the data source that you just created.

New Delete Test connection Manage state...						
   						
Select	Name ▾	JNDI name ▾	Scope ▾	Provider ▾	Description ▾	Category ▾
You can administer the following resources:						
☐	MyDatasourceName	jdbc/MyDatasouce	Cell=co-qa3Node02Cell	Oracle JDBC Driver (XA)	New JDBC Datasource	
Total 1						

14. Under **Related Items**, drill down on **JAAS - J2C authentication data**.

Test connection

General Properties

- * Scope: cells:co-qa3Node02Cell
- * Provider: Oracle JDBC Driver (XA)
- * Name: MyDatasourceName
- JNDI name: jdbc/MyDatasource
- ☒ Use this data source in container managed persistence (CMP)
- Description: New JDBC Datasource

Additional Properties

- Connection pool properties
- WebSphere Application Server data source properties
- Custom properties

Related Items

- JAAS - J2C authentication data

15. Ensure that the check box **Prefix new alias names with the node name of the cell (for compatibility with earlier releases)** is selected.

Data sources

Data sources > sl > JAAS - J2C authentication data

Specifies a list of user identities and passwords for Java(TM) 2 connector security to use.

☒ Prefix new alias names with the node name of the cell (for compatibility with earlier releases)

Apply

Preferences

New Delete

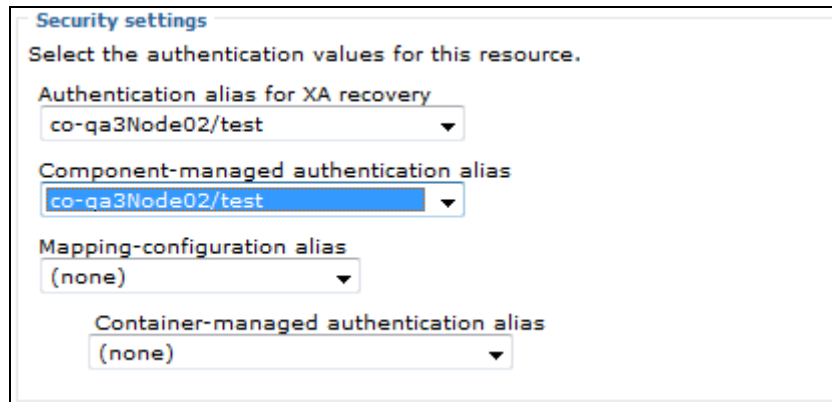
16. Click **New**.

17. Specify a unique alias.

18. The **User Id** and **Password** you specify should match the schema of the user who will log in to the database.

19. Click **OK**. Save the configuration.

20. Again, drill down on the data source that you just created.
21. Under **Security settings**, set **Authentication alias for XA recovery** to the authentication data that you just created and set **Component-managed authentication alias** to the same authentication data.



Security settings
Select the authentication values for this resource.

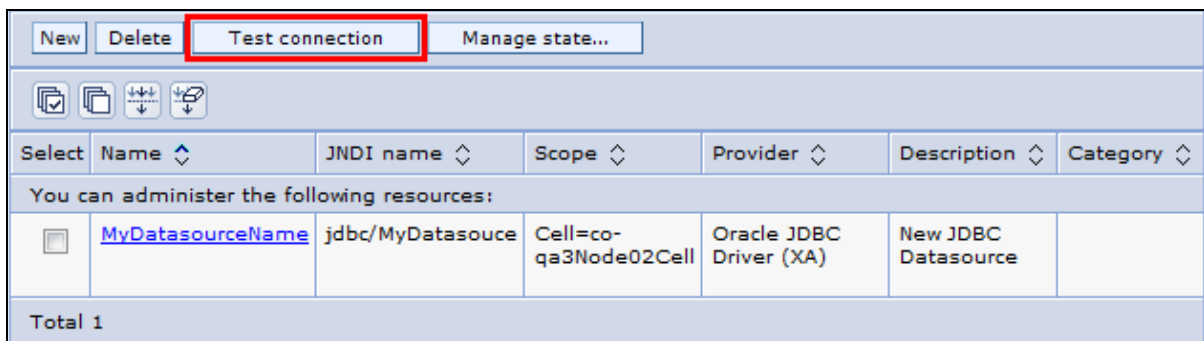
Authentication alias for XA recovery
co-qa3Node02/test

Component-managed authentication alias
co-qa3Node02/test

Mapping-configuration alias
(none)

Container-managed authentication alias
(none)

22. Click **OK**. Save the configuration.
23. Select the checkbox for the new data source. Click the **Test connection** button to verify that the data source can connect to the database.



New Delete **Test connection** Manage state...

☒ ☐ ☐ ☐

Select	Name	JNDI name	Scope	Provider	Description	Category
☐	MyDatasourceName	jdbc/MyDatasource	Cell=co-qa3Node02Cell	Oracle JDBC Driver (XA)	New JDBC Datasource	

Total 1

Setting Up JMS

CodeObjects Platform's EDA (Event-Driven Architecture) is implemented using JMS. Events of a certain type are published to a Topic identified by a JNDI name derived from the Event type. To setup EDA, you must:

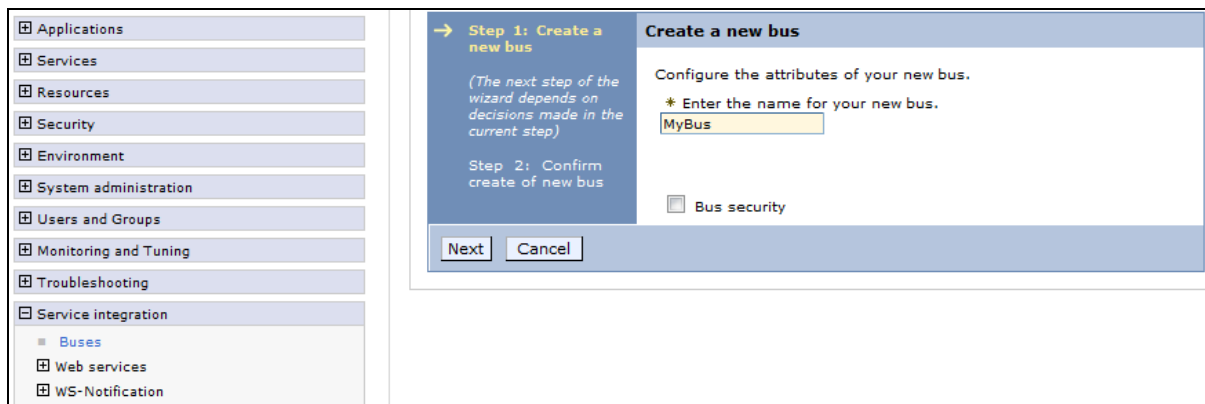
- Create one or more Integration Service Buses.
- Create one or more JMS Topic Connection Factories for each application with a JNDI name derived from the application's name.
- Create one or more Topics for each Event that can be raised by the application with a JNDI name derived from the application's name and the Event name.

These following steps describe the process in detail.

Create an Integration Service Bus

To create an integration service bus, from the WebSphere Integrated Solutions Console:

1. In the Tasks pane, expand **Service integration** and click **Buses**.
2. In the Configuration pane, click **New**.
3. Specify a unique name for the new bus.



4. Click **Next**.
5. When the summary of the new bus you are about to create is acceptable, click **Finish** to create it. Save the configuration.
6. Drill down on the newly created bus.

- Under **Topology**, drill down on **Bus members**.

The screenshot shows the WebSphere console interface. On the left, the 'General Properties' tab is active, displaying fields for 'Name' (TestBus), 'UUID' (E55D2A63CDC38E47), 'Description' (empty), and 'Inter-engine transport chain' (empty). On the right, the 'Topology' tab is active, showing a tree view with 'Bus members' highlighted in a red box. Below 'Bus members' are links for 'Messaging engines', 'Foreign bus connections', and 'Bootstrap members'. At the bottom right, under 'Destination resources', are links for 'Destinations' and 'Mediations'.

- Click **Add**, select the **Server** option button, and choose your server name or your cluster name.

The screenshot shows a wizard titled 'Add a server, cluster or a WebSphere MQ server as a new member of the bus.' The left sidebar indicates 'Step 1: Select server, cluster or WebSphere MQ server' is the current step, with 'Step 2: Summary' next. The main area is titled 'Select server, cluster or WebSphere MQ server' and contains the instruction 'Choose the server, cluster or WebSphere MQ server to add to the bus'. There are two radio button options: 'Server' (selected) and 'WebSphere MQ server'. The 'Server' option has a dropdown menu showing 'co-qa3Node02:server1'. The 'WebSphere MQ server' option has a dropdown menu showing '(none)'. At the bottom are 'Next' and 'Cancel' buttons.

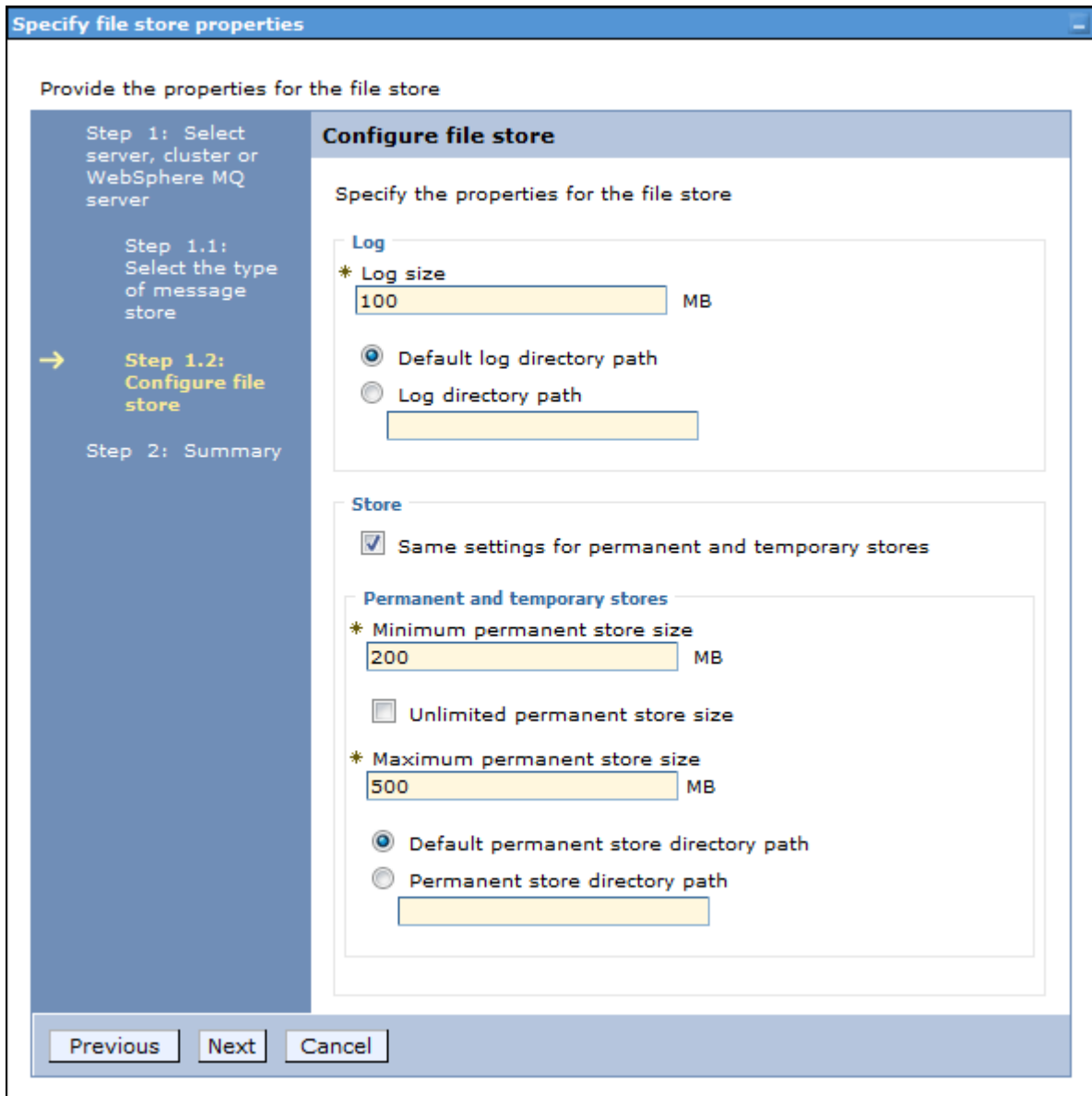
- Click **Next**.

10. For the type of **message store**, select the **File store** option button.

The screenshot shows a wizard window titled "Choose the type of message store for the persistence of message state". The window is divided into two main sections. On the left, a blue sidebar contains navigation links: "Step 1: Select server, cluster or WebSphere MQ server", "→ Step 1.1: Select the type of message store" (which is highlighted), and "Step 2: Summary". Below these links is a note: "(The next step of the wizard depends on decisions made in the current step)". The main area on the right has a title bar "Select the type of message store" and contains the same title text. Below this, there are two radio button options: "File store" (which is selected) and "Data store". At the bottom of the window, there are three buttons: "Previous", "Next", and "Cancel".

11. Click **Next**.

12. Under **Configure file store** accept the default settings.



The image shows a 'Specify file store properties' dialog box. On the left is a navigation pane with three steps: 'Step 1: Select server, cluster or WebSphere MQ server', 'Step 1.1: Select the type of message store', and 'Step 1.2: Configure file store' (which is highlighted with a yellow arrow). Below these is 'Step 2: Summary'. The main area is titled 'Configure file store' and contains the text 'Specify the properties for the file store'. It has two sections: 'Log' and 'Store'. The 'Log' section has a 'Log size' field set to '100' MB and two radio buttons: 'Default log directory path' (selected) and 'Log directory path' (with an empty text field). The 'Store' section has a checked checkbox for 'Same settings for permanent and temporary stores'. Below this is a 'Permanent and temporary stores' section with 'Minimum permanent store size' set to '200' MB, an unchecked checkbox for 'Unlimited permanent store size', 'Maximum permanent store size' set to '500' MB, and two radio buttons: 'Default permanent store directory path' (selected) and 'Permanent store directory path' (with an empty text field). At the bottom are 'Previous', 'Next', and 'Cancel' buttons.

Specify file store properties

Provide the properties for the file store

Step 1: Select server, cluster or WebSphere MQ server

Step 1.1: Select the type of message store

→ **Step 1.2: Configure file store**

Step 2: Summary

Configure file store

Specify the properties for the file store

Log

* Log size
100 MB

☒ Default log directory path
☐ Log directory path

Store

☒ Same settings for permanent and temporary stores

Permanent and temporary stores

* Minimum permanent store size
200 MB

☐ Unlimited permanent store size

* Maximum permanent store size
500 MB

☒ Default permanent store directory path
☐ Permanent store directory path

Previous Next Cancel

13. Click **Next**.

14. Under **Tune performance parameters** accept the default settings.

Improve messaging performance

Tune application server for messaging performance.

Step 1: Select server, cluster or WebSphere MQ server

Step 1.1: Select the type of message store

Step 1.2: Configure file store

→ **Step 1.3: Tune performance parameters**

Step 2: Summary

Tune performance parameters

To improve performance of messaging within the application server, the proposed Java Virtual Machine settings are advised. By default the initial and maximum JVM settings will remain unchanged, select the 'Change heap sizes' checkbox to modify the settings to the proposed values. On machines with low amounts of physical memory size or large numbers of application server instances, it may be necessary to reduce the proposed values accordingly.

☐ Change heap sizes

	Current heap sizes	Proposed heap sizes
Initial JVM heap size	512 MB	768 MB
Maximum JVM heap size	1024 MB	1024 MB

Previous Next Cancel

15. Click **Next**.

16. When the summary of the new bus you are about to create is acceptable, click **Finish** to create it. Save the changes.

Create Topic Connection Factory

There is one connection factory per CodeObjects Platform application instance.

The JNDI name for the connection factory has the following convention:

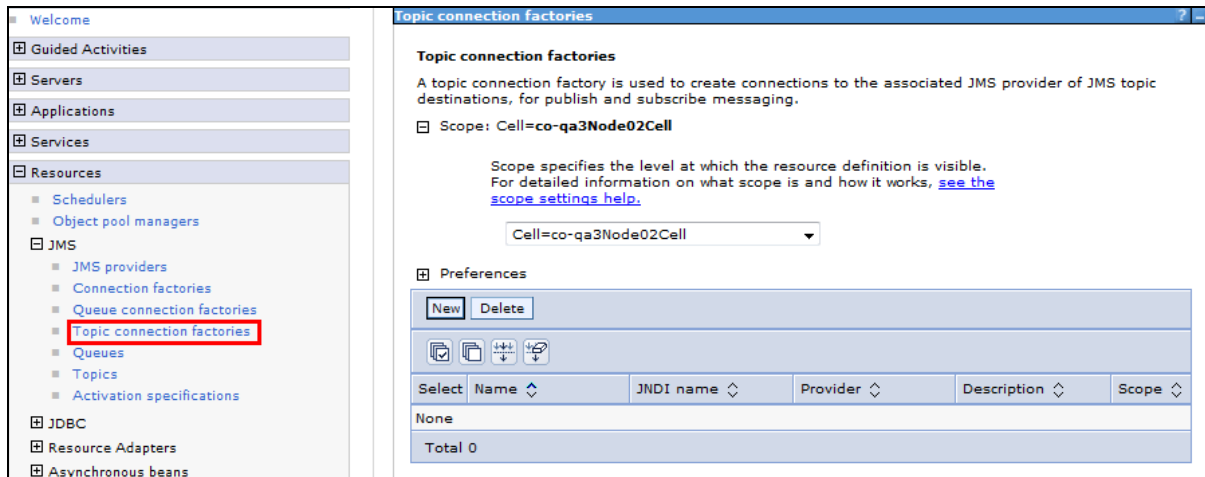
jms/{ApplicationName}/Connection Factory

Here, **{ApplicationName}** is the name of the application, that is, the name (without extension) of the application's *.application* file. For example, **acg-sl-policy-application-web-Websphere** is the name of the Specialty Line application and the JNDI name should be:

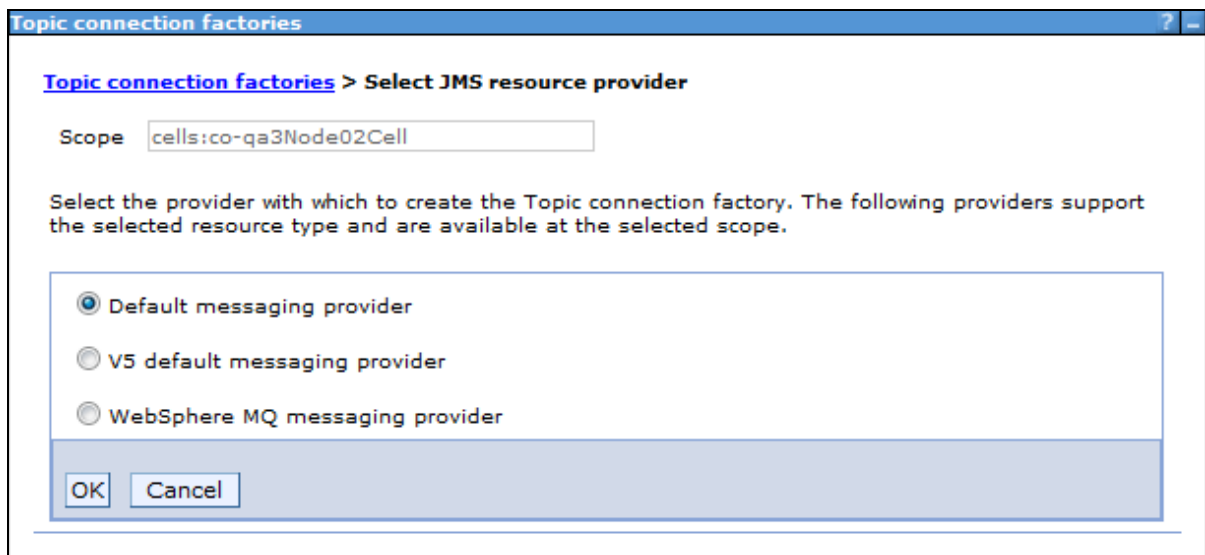
Jms/acg-sl-policy-application-web-Websphere/Connection Factory

Create the Connection Factory from the Websphere Integrated Solutions Console

1. In the Tasks pane, expand **Resources** → **JMS** and click **Topic connection factories**. Set the correct **Scope**.



2. In the Configuration pan, click **New**.
3. Select the **Default messaging provider** option button.



4. Click **OK**.

- Under **Name**, specify a unique topic connection factory name.

[Topic connection factories](#) > [Default messaging provider](#) > **New**

A JMS topic connection factory is used to create connections to the associated JMS provider of JMS topics, for publish/subscribe messaging. Use topic connection factory administrative objects to manage JMS topic connection factories for the default messaging provider.

Configuration

General Properties

Administration

Scope
Cell=co-qa3Node02Cell

Provider
Default messaging provider

* Name
TestConnectionFactory

* JNDI name
jms/testApplication/Connectio

Description

Category

Connection

* Bus name
TestBus

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

- Connection pool properties

Related Items

- JAAS - J2C authentication data
- Buses

- For **JNDI name**, specify **jms/application/Connection Factory**. Substitute **application** with the name of your application.
- For **Bus name**, select the bus that you created.
- Under **Durable subscription** section, specify a unique **Client identifier**.
- Click **OK** to create the topic connection factory. Save your changes.

Create Topics

For each event defined in the application, we must create a corresponding Topic.

The Topic is identified by a fixed JNDI name that is derived from the event definition. A service that raises these types of events will use a fixed JNDI name to look for the Topic to publish these events. A service that handles these types of events will use the same JNDI name to look for the Topic to subscribe to. It is developer's responsibility to document the events that an application can raise.

The JNDI name for an event in an application has the following convention:

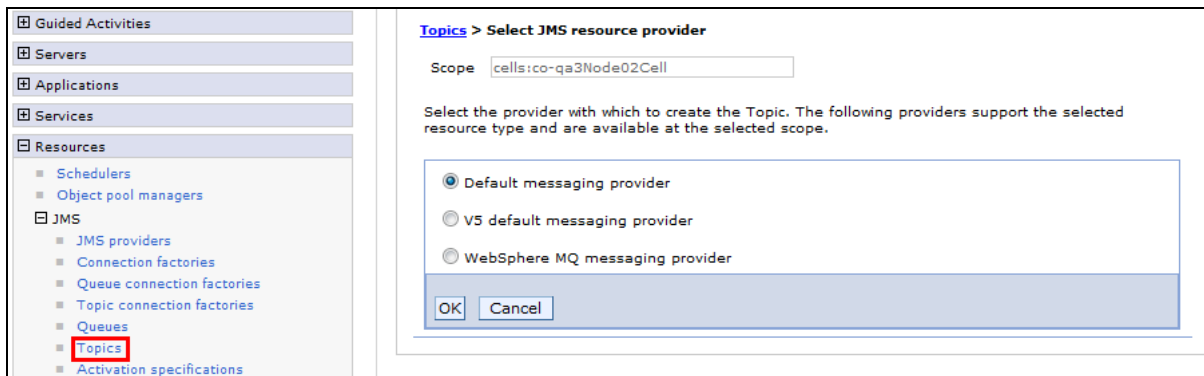
jms/{ApplicationName}/{EventName}

Here, **{ApplicationName}** is the name of the application, that is, the name (without extension) of the application's *.application* file. For example, **acg-sl-policy-application-web-Websphere** is the name of the Specialty Line application. **{EventName}** is the name of the event.

For the **AuditEvent** used in the **acg-sl-policy-application-web-Websphere** application, the JNDI name for the Topic should be:
jms/acg-sl-policy-application-web-Websphere/AuditEvent

Create the Topic from the WebSphere Integrated Solutions Console

1. In the Tasks pane, expand **Resources** → **JMS** and click **Topics**.
2. In the Configuration pane, select the proper server **Scope**.
3. Click **New**.
4. Select the **Default messaging provider** option button.



5. Click **OK**.

- For **Name**, specify a unique name.

The screenshot shows the 'General Properties' dialog box with two tabs: 'Administration' and 'Connection'. The 'Administration' tab is active, showing fields for 'Scope' (Cell=co-qa3Node02Cell), 'Provider' (Default messaging provider), 'Name' (TestAuditEvent), 'JNDI name' (jms/acg-sl-policy-application-), and 'Description'. The 'Connection' tab is also visible, showing fields for 'Topic name' (TestTopicName), 'Bus name' (TestBus), 'Topic space' (Default.Topic.Space), and 'JMS delivery mode' (Application).

- For the **JNDI name**, specify **jms/acg-sl-policy-application-web-WebSphere/AuditEvent**, assuming that you are defining the AuditEvent for the acg-sl-policy-application-web-WebSphere application.
- Under **Connection**, for **Topic name**, specify a unique name.
- For **Bus name**, specify the bus that you created.
- For **Topic space**, choose **Default.Topic.Space**
- Click **OK** to create the topic. Save the changes.
- Consult your documentation for a complete list of Topics for each of your application. Create all the Topics.
- Restart the Application Server.

JNDI Lookup for Service Properties

The *Service* metadata node has a sub-node called *Properties* that contains system properties and optionally their default values.

Solving a Problem

If you want to change a value in the *Properties* node, you must change the application's metadata files, which also means that you must rebuild and redeploy the WAR file.

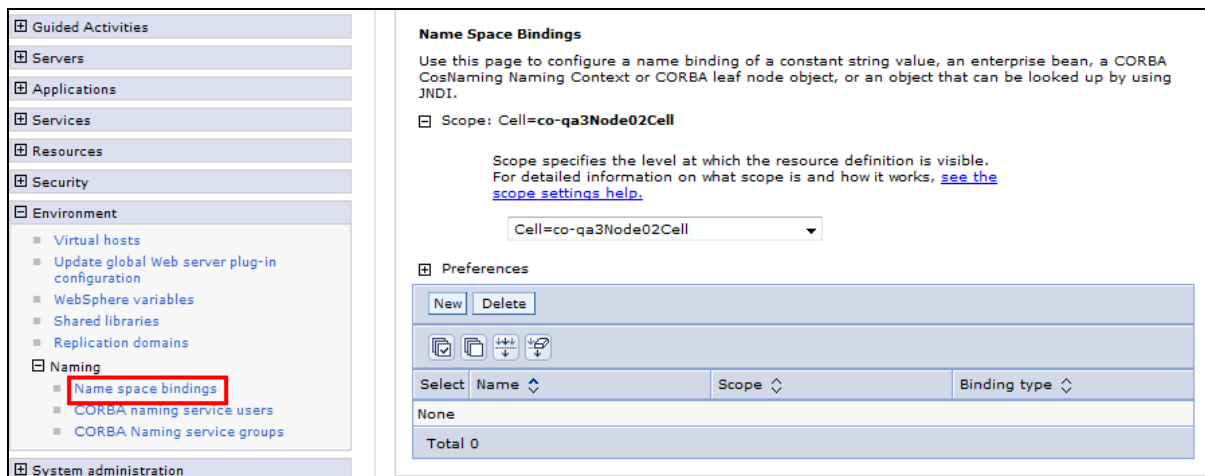
To solve this problem, you are allowed to define the *Properties* values in JNDI of the application server. If the values are found in JNDI, the values will be picked using JNDI. Otherwise, values default to the values defined in the *Properties* metadata node.

Thus, if you must change a value, you need only change the value of the JNDI entry, without the need to rebuild the war file.

The JNDI names follow a convention. For example, in application **acg-sl-policy-application-web-WebSphere**, for service instance **MyServiceImpl**, and for property name **myprop**, the JNDI name of the property will be:
co/acg-sl-policy-application-web-WebSphere/MyServiceImpl/myprop

To set this up, from the WebSphere Integrated Solutions Console:

1. In the Tasks pane, expand **Environment** → **Naming** and click **Name space bindings**. You can set the **Scope** to be **Cell**, **Cluster (In NetWork Deployment)** or **Server level** depending on your environment and business requirement.



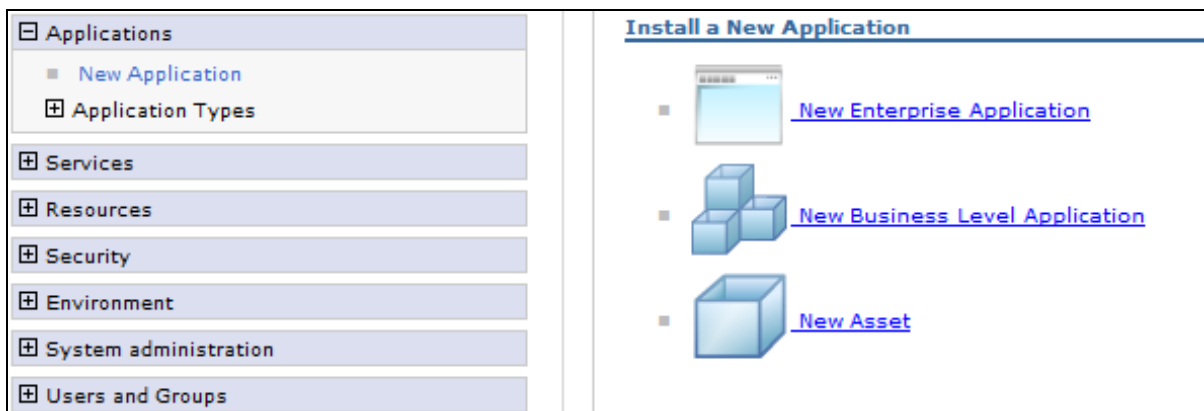
2. Click **New**, make sure the binding type is **String**, and click **Next**.

3. Provide the **Unique Binding Identifier, Name** in the namespace (for example: **co/policy-application-web-Websphere/WebDAVFileStorageServiceImpl/host**) and **String Value**.
4. Repeat the above for all the Service Properties that you want to configure. See your developer's documentation for a complete list of service properties for all applications.
5. Save your changes and restart the server.

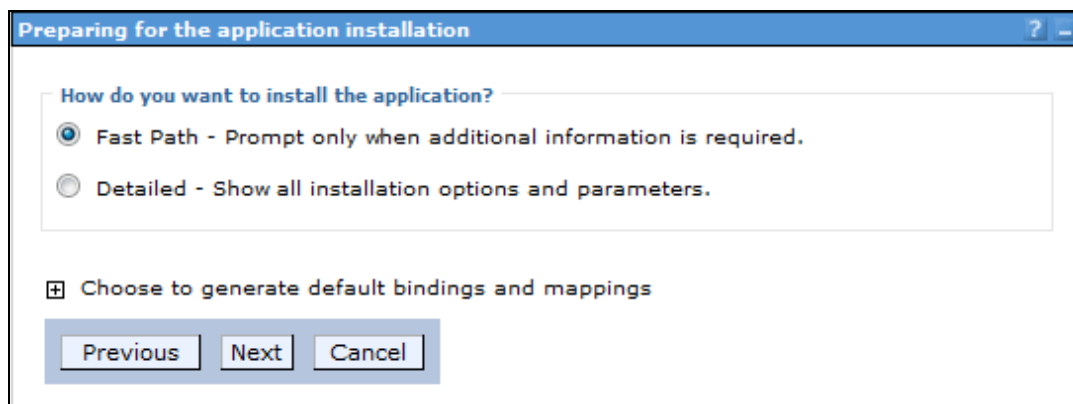
Deploy the Application

To deploy your application:

1. In the Tasks pane, expand **Applications** and click **New Application**.



- a. In the Configuration pane, drill down on **New Enterprise Application**.
- b. Browse to select the WAR file from either the local file system or a remote path. Click **Next**.
- c. Under **How do you want to install the application?** Select the option button **Fast Path – Prompt only when additional information is required**. Click **Next**.



- You may specify a new **Application name** if you like or leave the default name (the name of the WAR file). Click **Next**.

→ **Step 1: Select installation options**

[Step 2: Map modules to servers](#)

★ [Step 3: Map virtual hosts for Web modules](#)

★ [Step 4: Map context roots for Web modules](#)

[Step 5: Summary](#)

Select installation options

Specify the various options that are available to prepare and install your application.

☐ Precompile JavaServer Pages files

Directory to install application

☒ Distribute application

☐ Use Binary Configuration

☐ Deploy enterprise beans

Application name

- For the **Map modules to servers** step, click **Next**.

[Step 1: Select installation options](#)

→ **Step 2: Map modules to servers**

★ [Step 3: Map virtual hosts for Web modules](#)

★ [Step 4: Map context roots for Web modules](#)

[Step 5: Summary](#)

Map modules to servers

Specify targets such as application servers or clusters of application servers where you want to install the modules that are contained in your application. Modules can be installed on the same application server or dispersed among several application servers. Also, specify the Web servers as targets that serve as routers for requests to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated, based on the applications that are routed through.

Clusters and servers:

Select	Module	URI	Server
☐	TutorialApplicationWebClient.war	TutorialApplicationWebClient.war,WEB-INF/web.xml	WebSphere:cell=co-qa2Node01Cell,node=co-qa2Node02,server=server1

- For the **Map virtual hosts for Web modules** step, click **Next**.

[Step 1: Select installation options](#)

[Step 2: Map modules to servers](#)

→ **Step 3: Map virtual hosts for Web modules**

★ [Step 4: Map context roots for Web modules](#)

[Step 5: Summary](#)

Map virtual hosts for Web modules

Specify the virtual host where you want to install the Web modules that are contained in your application. You can install Web modules on the same virtual host or disperse them among several hosts.

☒ Apply Multiple Mappings

Select	Web module	Virtual host
☐	TutorialApplicationWebClient.war	default_host ▼

- For the **Map context roots for Web modules** step, specify a unique **Context root** and click **Next**.

Map context roots for Web modules

Context root defined in the deployment descriptor can be edited.

Web module	URI	Context Root
TutorialApplicationWebClient.war	TutorialApplicationWebClient.war, WEB-INF/web.xml	/TestContextRoot

Previous Next Cancel

- When the summary of the new enterprise application you are about to deploy is acceptable, click **Finish** to deploy. After the application has been installed, save the changes.

Set the Class loader

- In the Tasks pane, expand **Applications** → **Application Types** and click **WebSphere enterprise applications**.
- In the Configuration pane, drill down on the enterprise application that you just deployed.
- In the **Modules** section, drill down on **Manage Modules**.
- Drill down on the module that corresponds to the WAR file that you just deployed.
- For **Class loader order**, select **Classes loaded with local class loader first (parent last)**.

specific information for a Web module and session management settings.

Configuration

General Properties

- * URI: TutorialApplicationWebClient.war
- Alternate deployment descriptor:
- * Starting weight: 10000
- * Class loader order: **Classes loaded with local class loader first (parent last)**

Apply OK Reset Cancel

Additional Properties

- View Module Class Loader
- Target specific application status
- View Deployment Descriptor
- Session Management

- Click **OK**. Save the changes.

Updating Your Deployment

When redeploying an existing application, you must check the following:

1. If your application is accessing a different database, you must configure access to your database (data source).
2. If you have added any event definitions to your application, you must define a JMS topic for your event.
3. If you have changed your Application metadata.
4. If you have added any system properties, you must set up JNDI for any new properties.
5. Redeploy the WAR file.

In the following instructions, we assume that you are just *redeploying* your WAR file without making changes to your application that would require any of the other configuration parameters to change.

To configure the settings below, you must be logged in to the WebSphere Integrated Solutions Console in your browser.

To Redeploy Your Application

Typically, you should uninstall the existing application and reinstall it.

1. Click **Applications** → **New Application**.
 - a. Drill down on **New Enterprise Application**.
 - b. Browse to select the WAR file from either the local file system or a remote path. Click **Next**.
 - c. Answer the question **How do you want to install the application?** with **Fast Path**. Click **Next**.
2. You may specify a new **Application name** if you like or leave the name at the default (the name of the WAR file). Click **Next**.
3. On the **Map modules to servers** stage, click **Next**.
4. On the **Map virtual hosts for Web modules** stage, click **Next**.
5. On the **Map context roots for Web modules** stage, specify a unique context root and click **Next**.

6. When the summary of the new enterprise application you are about to redeploy is acceptable, click **Finish** to deploy.

Set the Class loader

Because the shared library is set up on a system basis, perform these steps *only* if the application is being set up with a different scope.

1. Click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Drill down on the enterprise application that you just deployed.
3. In the **Modules** section, drill down on **Manage Modules**.
4. Drill down on the module listed that corresponds to the WAR file that you just deployed.
5. For the **Class loader order**, specify **Classes loaded with local class loader first (parent last)**.
6. Click **OK**.