IBM WebSphere Commerce - Express Developer Edition

# Migration Guide

*Version 5.5*

IBM WebSphere Commerce - Express Developer Edition

# Migration Guide

*Version 5.5*

**Note:**
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 69.

**First Edition, First Revision (September, 2004)**

This edition applies to version 5.5 of IBM WebSphere Commerce - Express Developer Edition for Windows 2000 (product number 5724-A18) and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality.

IBM welcomes your comments. You can send your comments by using the online IBM WebSphere Commerce documentation feedback form, available at the following URL:

http://www.ibm.com/software/webservers/commerce/rcf.html

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

# Preface

This book describes how to migrate the WebSphere® Commerce development environment from WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5. In addition, if you have applied any of the fix packs or Commerce Enhancement Packs for WebSphere Commerce, Version 5.4 and have used WebSphere Commerce Studio, Version 5.4 with these versions of WebSphere Commerce, migration from those code levels are also supported.

## Updates to this book

This migration guide, and any updated versions of this migration guide, will be available in the WebSphere Commerce Technical Library.

To learn about last-minute changes to the product, see the updated product README file, also available from the above Web site. If you are installing the the WebSphere Commerce development environment on a machine that does not have WebSphere Commerce Studio, Version 5.4 and the supported products installed, see the *WebSphere Commerce Studio Installation Guide* for your database.

+ Updates from the last version of this book are identified by revision characters
+ contained in the margin. This book uses the following conventions for
+ revision characters:

+ • The "+" (plus) character identifies updates that have been made in the
+ current version of this book.

+ • The "|" (split vertical bar) character identifies updates that have been made
+ in the previous versions of this book.

+ The following table shows the main changes that have been made to this
+ book.

‡

| Change | Chapter, section, or page affected |
|---|---|
| + Updated the instructions to export multiple EJB groups. + | + See "If you have many custom enterprise beans" on page 24. + |

## Conventions used in this book

This book uses the following highlighting conventions:

- **Boldface type** indicates commands or graphical user interface (GUI) controls such as names of fields, icons, or menu choices.
- `Monospace type` indicates examples of text you enter exactly as shown, file names, and directory paths and names.
- *Italic type* is used to emphasize words. Italics also indicate names for which you must substitute the appropriate values for your system. When you see any of the following names, substitute your system value as described:

  *host_name*
  The fully qualified host name of your the WebSphere Commerce development environment machine (for example, ibm.com is fully qualified).

  *drive* The letter representing the drive on which you installed the product or component being discussed (for example, C:).

  *WC_installdir*
  The installation path for WebSphere Commerce - Express 5.5. The default installation path is \Program Files\WebSphere\CommerceServer55. For example, c:\Program Files\WebSphere\CommerceServer55.

  *WCDE_installdir*
  The installation path for the the WebSphere Commerce development environment. The default installation path is \WebSphere\CommerceDev55. For example, c:\WebSphere\CommerceDev55.

  *WSAD_installdir*
  The installation path for WebSphere Studio Application Developer, Version 5.1. The default installation path is \WebSphereStudio. For example, c:\WebSphereStudio.

  *WSAD_workspace*
  The installation path for your WebSphere Studio Application Developer, Version 5.1 workspace. The default installation path is \WebSphere\workspace_db2. For example, c:\WebSphere\workspace_db2.

This icon marks a Tip - additional information that can help you complete a task.

## Where to find more information

For information related to the WebSphere Commerce development environment, refer to the WebSphere Commerce development environment site.

For information on WebSphere Commerce, refer to the following Web sites:

- WebSphere Commerce Technical Library, which includes a section on downloading updated documentation, such as guides and the online help.
- WebSphere Commerce Support
- WebSphere Commerce - Express overview

# Part 1. Pre-migration

This part discusses important information you should review before you migrate to the WebSphere Commerce development environment. Information about hardware and prerequisites, setup and possible migration environments, and important code changes are included in this section.

# Chapter 1. Before you migrate your WebSphere Commerce development environment

This chapter provides information on what you should note and do before you migrate from WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5.

## Prerequisites for migration

To use the migration instructions in this guide, your WebSphere Commerce Studio, Version 5.4 machine must meet the hardware and software requirements detailed in the following sections.

### Prerequisite hardware

- You require a dedicated Pentium® III 733 MHz IBM-compatible personal computer with the following:
  - A minimum of 1.5 GB of random access memory (RAM).
  - The following amounts of disk space for each option that you choose to install:
    - For store front asset development:
      - WebSphere Studio — 300 MB
      - Store Archive Tools — 25 MB
      - Blaze Advisor and Blaze Innovator Workbench — 55 MB
    - For back-end development:
      - WebSphere Application Developer for Java™ Enterprise — 1.2 GB
      - IBM® DB2® Universal Database — 300 MB
      - WebSphere Commerce development environment — 300 MB
      - Blaze Advisor™ Server — 45 MB
    - IBM Distributed Debugger — 160 MB
    - Applet Designer — 15 MB
    - Page Detailer — 10 MB

    You will also need an additional 500 MB on the `C:` drive. If your machine is formatted with FAT partitioning and the partition is over 1.024 GB, you will need twice as much free disk space. The installation will check for adequate free disk space and will warn you if there is not enough space.
  - A CD-ROM drive.

- A graphics-capable monitor with a color depth of at least 256 colors, with screen resolution set to 800x600 or higher.
- A mouse or other pointing device.
- A local area network (LAN) adapter that is supported by the TCP/IP protocol.

### Prerequisite software

Ensure that you meet the following minimum software requirements before migrating to WebSphere Commerce - Express Developer Edition, Version 5.5:

- Ensure that you have Windows® 2000 Server Edition, or Advanced Server Edition installed, with Service Pack 3 applied. You can obtain updates from http://www.microsoft.com.

  **Important**

  If you are using Windows NT® with WebSphere Commerce Studio, Version 5.4, you must upgrade your operating system to Windows 2000 before migrating to WebSphere Commerce - Express Developer Edition, Version 5.5. This version of the WebSphere Commerce development environment is not supported on Windows NT.

- Install Internet Explorer 5.5 Service Pack 1 with the latest critical security updates, on each machine that you will use to access WebSphere Commerce Studio. A copy of Microsoft® Internet Explorer is available by way of download on the Microsoft Web site at http://www.microsoft.com.
- After you install the WebSphere Commerce development environment, you need to install the IBM WebSphere Commerce 5.5.0.2 fix pack. This fix pack includes fixes and enhancements for the migration process. This fix pack is available from the WebSphere Commerce Support site. Click on the link for the edition you are using and see the section Software downloads. Download this fix pack, and follow the instructions that are provided to install it. (If you plan to upgrade from WebSphere Studio Application Developer, Version 5.1 to WebSphere Studio Application Developer, Version 5.1, instructions are also provided in the fix pack documentation.) You should not proceed with the migration steps in this guide (such as migrating your instance or database) without installing this fix pack.

## Migration environments

When you migrate from WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5, you can select to migrate to an environment that is comparable to that if you completed a new installation of the WebSphere Commerce development environment. That

is, you can remove the unneeded WebSphere Commerce assets from your machine, after the WebSphere Commerce development environment migration is complete.

You are not required to remove these assets if you wish to continue having a local instance of WebSphere Commerce. In this case, it is recommended that you migrate your local instance of WebSphere Commerce to the WebSphere Commerce 5.5 level.

The following environments are supported as the end result of migrating the WebSphere Commerce development environment:

- Environment A — Pure development environment
- Environment B — Development environment and WebSphere Commerce coexist but do not share assets.

The migration paths for these scenarios are described in the following sections. In each case, the environments can use either a remote WebSphere Commerce database, or a local (stand-along configuration) database. In addition, each environment can be used for developing storefront assets (such as the look and feel of a store), store back office logic (such as custom commands and enterprise beans), or a combination of both storefront assets and back office logic.

The high-level steps to migrate to WebSphere Commerce - Express Developer Edition, Version 5.5 using either environment A or B are as follows:

__ 1. Ensure you have reviewed and understood the information described in “Prerequisites for migration” on page 3 and “Migration environments” on page 4.

__ 2. Back up your existing WebSphere Commerce Studio, Version 5.4, and WebSphere Commerce 5.4 assets as outlined in “Backing up your WebSphere Commerce and the WebSphere Commerce development environment assets” on page 21.

__ 3. Upgrade to DB2 - ExpressVersion 8.1 s outlined in “Upgrading your DB2 product level” on page 21.

__ 4. Upgrade WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5. The detailed installation and upgrade steps are described in “Upgrading the WebSphere Commerce development environment” on page 22. This includes the installation of WebSphere Studio Application Developer, Version 5.1.

__ 5. Install the IBM WebSphere Commerce 5.5.0.2 fix pack. This fix pack includes fixes and enhancements for the migration process. This fix pack is available from the WebSphere Commerce Support site. Click on the link for the edition you are using and see the section Software

downloads. Download this fix pack, and follow the instructions that are provided to install it. You should not proceed with the migration steps in this guide (such as migrating your instance or database) without installing this fix pack.

__ 6. Migrate the existing WebSphere Commerce Studio, Version 5.4 database schema to the WebSphere Commerce - Express Developer Edition, Version 5.5 level as described in "Migrating the WebSphere Commerce development environment database schema" on page 25.

__ 7. Migrate the existing WebSphere Commerce instance to the WebSphere Commerce 5.5 level as described in "Migrating the WebSphere Commerce instance" on page 26.

**Note:** Instance migration involves automatically copying the store front or Web assets (that is, JSP files, HTML pages, or graphics) from VisualAge® for Java to WebSphere Studio Application Developer. However, you should also ensure that you have accounted for all the assets created for your store front within WebSphere Commerce Studio, Version 5.4.

Migrating the existing WebSphere Commerce instance to the WebSphere Commerce 5.5 level includes the following:

- Migrating the instance.xml file.
- Copying the store front or Web assets from VisualAge for Java into a WebSphere Studio Application Developer Web project. This includes any JSP files, HTML pages, or graphics you used with your store front.
- Extracting the JAR file containing the non-enterprise bean custom code into the WebSphere Studio Application Developer Java project called WebSphereCommerceServerExtensionsLogic.
- Migrating the JSP files that were copied when you run the instance migration script, to conform to the JSP 1.2 specification.

__ 8. Transition your WebSphere Commerce customized code contained in VisualAge for Java to WebSphere Studio Application Developer, Version 5.1. The detailed steps are described in "Transitioning your customized or extended code" on page 30.

__ 9. Migrate the WebSphere Commerce payment database and instance as outlined in "Migrating the payment instance and database" on page 41. For WebSphere Commerce 5.4, the payment database was required to be installed on a different machine than the one containing WebSphere Commerce. For WebSphere Commerce 5.5, you have the option of installing the payment database locally on the same machine as the WebSphere Commerce development environment.

__ 10. Complete the post-migration steps outlined in Chapter 4, "Post-migration actions," on page 45. This includes uninstalling some components of WebSphere Commerce.

### Environment A — Pure development environment

In the case of a migration on a pure development environment, the WebSphere Commerce development environment is used for developing store assets and logic; it is not used to host a production store. The end result of such a migration is a similar structure to a brand new installation of the WebSphere Commerce development environment. That is, in a pure development environment, WebSphere Commerce does not exist on your machine after the migration and only the WebSphere Commerce development environment exists on the machine. In this case, your existing development store is migrated, and the FashionFlow sample store is installed.

### Environment B — Development environment and WebSphere Commerce coexist but do not share assets

In the case of a development environment and WebSphere Commerce coexisting, but not sharing assets, the end result should be one machine housing a development environment that is completely separated from the local instance of WebSphere Commerce 5.5. That is, in this environment, WebSphere Commerce and the WebSphere Commerce development environment both exist on one machine. Both the development environment and WebSphere Commerce uses a database; the databases are identical in content, but are not shared. After migration is complete, each of the databases should continue to be the same, as should the XML files. Note, that you are not required to migrate the database twice. Instead, after migrating the WebSphere Commerce development environment, you should back up the already migrated database, then create a new database (for example, `Mall2`) and restore to the new database. For details on this, refer to step 3 on page 46.

# Chapter 2. Code changes affecting your the WebSphere Commerce development environment migration

Once you have understood the prerequistes for migration and the possible migration environments, you need to review the code changes from WebSphere Commerce Studio, Version 5.4 that affect your migration. This chapter discusses the code changes you should note before migrating the WebSphere Commerce development environment.

## WebSphere Commerce 5.4 and WebSphere Commerce Studio 5.4 setup environment

With WebSphere Commerce Studio, Version 5.4, you had the option of installing the WebSphere Commerce development environment and WebSphere Commerce on the same machine so that they coexisted. You also had the option of installing a sample store that is used in the WebSphere Test Environment when you installed VisualAge for Java. If you selected the sample store, the following items were created:

- A database; for example, the default VAJ_DEMO database. You have the option to change this during the installation.
- The store directory structure.
- The instance XML file; for example, the default vaj_demo.xml instance XML file. You have the option to change this during the installation.

These assets are used by the WebSphere Test Environment. The available sample store for WebSphere Commerce Studio, Version 5.4 was the InFashion store and the the WebSphere Commerce development environment installation program used a master XML file to populate the sample store in the database.

Alternatively for WebSphere Commerce Studio, Version 5.4, you can maintain the WebSphere Commerce development environment and WebSphere Commerce on separate machines.

## Request for Quotation (RFQ)

A Request For Quotation (RFQ) is a process where a buying organization solicits offers from selling organizations to obtain a suitable price for a given product or set of products. The buyers in the buying organization may not find the products they wanted in the catalog of the selling organizations. In WebSphere Commerce 5.5, the RFQ request tools have been changed to allow the buyer to create an RFQ on a ″made to order″ item. To fully specify the attributes of the ″made to order″ item, personalization attributes are used just

like any other RFQ item. For details on RFQ migration, including new or changed (command or database table) assets and how to migrate the RFQ request tool, refer to ″Request For Quotation (RFQ) migration″ section within the *WebSphere Commerce Migration Guide*.

In addition, for information on supported, deprecated, reserved, and withdrawn API, refer to the WebSphere Commerce 5.5 Development online help.

## Collaborative workspaces

In WebSphere Commerce 5.4, the collaborative workspaces functionality were provided by the CollabManagerBean enterprise bean. In WebSphere Commerce 5.5, this enterprise bean has been replaced by the class CollabManager. If you have customized code that makes use of CollabManagerBean enterprise bean, do the following to migrate your code to work with the new CollabManager class:

1. Edit the class or package importing statement to reference the new CollabManager class as follows:
   a. Remove the import statement in you code that refers to the CollabManagerBean class (for example, `import com.ibm.commerce.collaboration.objects.CollabManagerBean;`).
   b. Insert a new import statement that referse to the CollabManager class (for example, `import com.ibm.commerce.collaboration.manager.*;`).
2. Edit the declare and instance statements to use the CollabManager type.
3. Edit the methods that are invoked by the object according to the API mapping tables below. You may be required to add additional code depending on how you map the new API.

   **Note:** The parameter or return type of the equivalent API may not be the same as in CollabManagerBean. Check the CollabManager class API reference for detail.

   The following table illustrates the API mapping between CollabManagerBean class and CollabManager class (note that the the interfaces that are required by the enterprise bean specification are not listed):

*Table 1. API mapping between CollabManagerBean class and CollabManager class*

| Method in CollabManagerBean | Equivalent method in CollabManager class |
|---|---|

*Table 1. API mapping between CollabManagerBean class and CollabManager class (continued)*

| | |
|---|---|
| addAuthorGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• addCWMembers(CollabWorkspaceInfo csbean)<br>• addCWMembers(String collabWorkspaceId,Vector memberDNs,String role)<br>• addCWMember(String collabWorkspaceId,String memberDN,String role) |
| ddManagerGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• addCWMembers(CollabWorkspaceInfo csbean)<br>• addCWMembers(String collabWorkspaceId,Vector memberDNs,String role)<br>• addCWMember(String collabWorkspaceId,String memberDN,String role) |
| addReaderGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• addCWMembers(CollabWorkspaceInfo csbean)<br>• addCWMembers(String collabWorkspaceId,Vector memberDNs,String role)<br>• addCWMember(String collabWorkspaceId,String memberDN,String role) |
| createCollabSpace (CollabSpaceBean cspaceBean, java.lang.String creatorDN) | createCollabWorkspace(CollabWorkspaceInfo cspaceBean) |
| deleteAuthorGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• removeCWMembers(CollabWorkspaceInfo csbean) removeCWMembers(String cwId, Vector memberDNs)<br>• removeCWMember(String cwId, String memberDN) |
| deleteCollabSpace (java.lang.String collabSpaceID) | removeCollabWorkpace(String collabSpaceID) |

*Table 1. API mapping between CollabManagerBean class and CollabManager class (continued)*

| | |
|---|---|
| deleteManagerGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• removeCWMembers(CollabWorkspaceInfo csbean)<br>• removeCWMembers(String cwId, Vector memberDNs)<br>• removeCWMember(String cwId, String memberDN) |
| deleteReaderGroupMember (CollabSpaceBean csbean) | Any of the following:<br>• removeCWMembers(CollabWorkspaceInfo csbean)<br>• removeCWMembers(String cwId, Vector memberDNs)<br>• removeCWMember(String cwId, String memberDN) |
| deleteCollabSpace (java.lang.String collabSpaceID) | getCollabWorkspaceDetails(String collabSpaceID) |
| isUserDNAuthorGroupMember (java.lang.String userDN, java.lang.String collabSpaceID) | getCWMemberRole(String memberDN, String cwId) |
| getCWMemberRole (String memberDN, String cwId) | getCWMemberRole(String memberDN, String cwId) |
| isUserDNReaderGroupMember (java.lang.String userDN, java.lang.String collabSpaceID) | getCWMemberRole(String memberDN, String cwId) |
| listAllCollabSpaces() | listCollabWorkspaces() |
| listAuthorGroupMembers (java.lang.String collabSpaceID) | listCWMembers(String collabSpaceID, String role) |
| listCollabSpaceForUserDN (java.lang.String userDN) | listCollabWorkspaces(String userDN) |
| listCollabSpaceTemplates() | listCWTemplates() |
| listCWTemplates() | listCWTemplates() |
| listReaderGroupMembers (java.lang.String collabSpaceID) | listCWMembers(String collabSpaceID, String role) |
| modifyAuthorGroupMembers (CollabSpaceBean csbean) | setCWMemberRole(String memberDN, String cwId, String role) |
| modifyCollabSpaceDescription (java.lang.String collabSpaceID, java.lang.String description) | changeCWDescription(String collabSpaceID,String description) |

*Table 1. API mapping between CollabManagerBean class and CollabManager class (continued)*

| | |
|---|---|
| changeCWDescription (String collabSpaceID,String description) | changeCWDescription(String collabSpaceID,String description) |
| changeCWDescription (String collabSpaceID,String description) | setCWMemberRole(String memberDN, String cwId, String role) |
| TESTaddAuthorGroupMembers (java.lang.String csID) | No applicable; no equivalent |
| TESTaddAuthorGroupMembers (java.lang.String csID, java.lang.String userDN) TESTaddManagerGroupMembers (java.lang.String csID) | No applicable; no equivalent |
| TESTaddManagerGroupMembers (java.lang.String csID, java.lang.String userDN) | No applicable; no equivalent |
| TESTaddReaderGroupMembers (java.lang.String csID) TESTaddReaderGroup Members(java.lang.String csID, java.lang.String userDN) | No applicable; no equivalent |
| TESTcreateCollabSpace (java.lang.String bProcType, java.lang.String bProcID) | No applicable; no equivalent |
| TESTdeleteAuthorGroupMembers (java.lang.String csID) | No applicable; no equivalent |
| TESTdeleteAuthorGroupMembers (java.lang.String csID, java.lang.String userDN) | No applicable; no equivalent |
| TESTdeleteManagerGroupMembers (java.lang.String csID) | No applicable; no equivalent |
| TESTdeleteManagerGroupMembers (java.lang.String csID, java.lang.String userDN) | No applicable; no equivalent |
| TESTdeleteReaderGroupMembers (java.lang.String csID) | No applicable; no equivalent |
| TESTdeleteReaderGroupMembers (java.lang.String csID, java.lang.String userDN) | No applicable; no equivalent |
| TESTmodifyAuthorGroupMembers() | No applicable; no equivalent |
| TESTmodifyManagerGroupMembers() | No applicable; no equivalent |

*Table 1. API mapping between CollabManagerBean class and CollabManager class (continued)*

| | |
|---|---|
| TESTmodifyReaderGroupMembers() | No applicable; no equivalent |

The following table lists other miscellaneous API mappings for collaborative workspaces:

*Table 2. Other API mappings for collaborative workspaces*

| **Method in CollabManagerBean** | **Equivalent implementation** |
|---|---|
| getCollabAppServerIP() | WcsApp.configProperties.getValue (ECConstants.EC_COLLAB_QP_HOST) |
| getCollabAppServerPort() | WcsApp.configProperties.getValue (ECConstants.EC_COLLAB_QP_HTTP_PORT) |

4. Since the CollabManager is not an enterprise bean, remove all the try and catch entries previously used to for enterprise bean coding to handle exceptions raised by enterprise beans. Remove the try and catch entries to avoid compling errors.
5. Compile your code and fix any compiling errors that are raised by your IDE.

## Database schema differences

For details on the database schema information for WebSphere Commerce 5.5 and the WebSphere Commerce development environment, refer to the WebSphere Commerce Development online help. Once you have launched the online help, select **WebSphere Commere Development information** > **Reference** > **Data** > **Database schema**. From here, select **Database tables** to view an alphabetical list of all database schema information. Select **Database changes in this release** to view the schema changes for Version 5.5. The online help also provides data model information.

Ensure that you visit the WebSphere Commerce Technical Library (http://www.ibm.com/software/commerce/library/), for the latest versions of the WebSphere Commerce documentation, including any updates to the database schema information in the online help.

## API differences

For details on the API information for WebSphere Commerce 5.5 and the WebSphere Commerce development environment, refer to the WebSphere Commerce Development online help. Once you have launched the online help, select **WebSphere Commere Development information** > **Reference** > **API information**. The online help also provides API information on the following:

- Which API are supported for Version 5.5.
- Which API have been changed for Version 5.5. If you used or customized any of these API, ensure that you review the changed versions. With WebSphere Commerce 5.5 and the WebSphere Commerce development environment, you will have migrated to enterprise bean level 1.1. For more information on this and enterprise bean Finder Helper implementation, refer to "Migrating enterprise beans from 1.0 to 1.1 specification" and "Differences between VisualAge for Java and WebSphere Studio Application Developer" on page 30.
- Which API have been reserved for IBM internal use and are for reference purposes only.
- Which API have been deprecated for Version 5.5. If you have used any of the deprecated API, refer to the online help for the replacement API to use with Version 5.5.
- Which API have been withdrawn from Version 5.5 and are no longer supported.
- Which API are no longer compatible with Version 5.5, but were supported for Version 5.1 and 5.4.

Cross reference information is also inlcude for data beans, enterprise beans, and database tables; and commands, tasks, and database tables.

Ensure that you visit the WebSphere Commerce Technical Library (http://www.ibm.com/software/commerce/library/), for the latest versions of the WebSphere Commerce documentation, including any updates to the API information the online help.

## Migrating enterprise beans from 1.0 to 1.1 specification

Transitioning from 1.0 to 1.1 specification does not impact the access bean layer or your code; you can use the code you used previously, with the WebSphere Commerce development environment. Transitioning from 1.0 to 1.1 does impact your enterprise beans. The following table illustrates the enterprise bean changes that were made for the WebSphere Commerce development environment, and thus will impact your migration from 1.0 ot 1.1 specification.

*Table 3. Enterprise bean changes and impact to the WebSphere Commerce development environment*

| Change made for the WebSphere Commerce development environment | How the change impacts your enterprise beans |
|---|---|
| Rename the enterprise bean ″impl″ to ″bean″ base class | No impact as this is a change that involves naming and packaging conventions. |
| Remove ″implements java.io.Serializable″ from BeanBase class | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration. |
| Remove serial version UID | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration. |
| Ensure ejbCreate() returns primary key object | You must make this change to comply with the enterprise bean 1.1 specification. |
| Remove RemoteException from ejbCreate(), ejbPostCreate(), ejbActivate(), ejbPassivate(), ejbStore(), ejbRemove(), and all remote methods | You must make this change to comply with the enterprise bean 1.1 specification. |
| Remove getEntityContext(), setEntityContext(), unsetEntityContext() from the bean or bean base class | No impact as this is an internal cleanup fix. |
| Move all ejbCreate() and ejbPostCreate() code to the bean base class | No impact as this is an internal cleanup fix. |
| Match ejbPostCreate() and ejbCreate() methods | You must make this change to comply with the enterprise bean 1.1 specification. |
| Add the WCSecurity role | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration.<br><br>The enterprise bean security role is needed if security is turned on within WebSphere Studio Application Developer. |

*Table 3. Enterprise bean changes and impact to the WebSphere Commerce development environment (continued)*

| | |
|---|---|
| Specify the enterprise bean transaction setting as ″Required″ | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration.<br><br>This enterprise bean transaction setting is required. The value depends on the enterprise bean usage. |
| Specify the enterprise bean isolation level as ″Repeatable Read″ for each enterprise bean | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration.<br><br>This enterprise bean isolation level is required. The value depends on the database type used. |
| Change the JNDI name | No impact as this is a change that involves naming and packaging conventions. |
| Delete xyzBeanFinderHelper interface | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration.<br><br>The BeanFinderHelper interface is obsolete. All finders are defined in ibm-ejb-jar-ext.xmi. |
| Rename xyzBeanFinderObjectImpl to xyzBeanFinderObjectBase | Although this change is not directly related to the enterprise bean 1.1 specification, you will need to make this change for your enterprise beans to work properly after migration.<br><br>The BeanFinderHelper interface is obsolete. All finders are defined in ibm-ejb-jar-ext.xmi. |

## Checking the unique index in the MSGTYPES table

The MSGTYPE_ID and NAME column are new for the MSGTYPES table. The MSGTYPE_ID column is a primary key and the NAME column is a unique index. If you have created your own message types, ensure that they do not conflict with the new ones. If they do, you can modify the MSGTYPE_ID value by adding 10,000 to the numbers or modify the NAME value. You will need to re-compile your commands that reference to the custom message types. Otherwise, you may encounter problems when migrating your data from from WebSphere Commerce 5.4

## WebSphere Application Command Framework

For WebSphere Commerce 5.4, the San Francisco Command Framework was used. WebSphere Commerce 5.5, however, has changed to use the WebSphere Application Command Framework. Thus, when migrating custom or extended code, it is important to note the following:

- All the customer commands that extend from WebSphere Commerce 5.4 controller commands or task command are required to be recompiled in order to pick up the new WebSphere Application Server, Version 5 runtime libraries.
- The callers of the controller commands and task commands are required to have the command context set.

## J2EE Connector Architecture

For WebSphere Commerce Suite 5.1 and WebSphere Commerce 5.4, the Common Connector Framework was used. WebSphere Commerce 5.5, however, has changed to use the J2EE Connector Architecture. If you used a connector with WebSphere Commerce Suite 5.1 or WebSphere Commerce 5.4, you will need to rewrite it. For details on J2EE framework and the J2EE Connector Architecture, refer to the follwing Web site from Sun Microsystem's: http://java.sun.com/j2ee/connector.

In addition, the WebSphere Commerce development environment also provides an example connector called the Sample Adapter, which is a sample J2EE Connector Architecture adapter (connector).

# Part 2. Migration flow

This part describes a typical migration flow to migrate from WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5. The migration process involves a series of steps. Refer the following chapters within this part for instructions on completing each migration step.

# Chapter 3. Migrating the WebSphere Commerce development environment

This chapter describes the detailed steps to upgrade the WebSphere Commerce development environment to WebSphere Commerce - Express Developer Edition, Version 5.5.

**Note:** Before you upgrade the WebSphere Commerce development environment, ensure that you have reviewed the information outlined in Part 1, "Pre-migration," on page 1, including the hardware and software requirements outlined.

## Backing up your WebSphere Commerce and the WebSphere Commerce development environment assets

Before you upgrade any software or transition any custom code to the WebSphere Commerce development environment level, ensure you back up both WebSphere Commerce and the WebSphere Commerce development environment assets, as follows:

- Back up the following WebSphere Commerce assets:
  - WebSphere Commerce directory tree
  - WebSphere Commerce database

  For the steps to back up the WebSphere Commerce directory tree and database, see the section "Backing up WebSphere Commerce 5.4" in the *WebSphere Commerce Migration Guide* for the version of WebSphere Commerce that you are migrating from.
- Back up the WebSphere Commerce development environment assets.

## Upgrading your DB2 product level

If you use DB2 as your database management system, after you have backed up your WebSphere Commerce and WebSphere Commerce Studio assets, you must upgrade to and configure DB2 - Express Version 8.1 before upgrading to the WebSphere Commerce development environment. To install and configure DB2 Version 8.1 as the WebSphere Commerce Studio database, follow these high-level steps:

1. Record the configuration settings for your database before DB2 migration.
2. Change the diagnostic error level.
3. Take the DB2 server offline for DB2 migration.
4. Verify that databases are ready for DB2 migration.

5. Back up your databases.
6. *Optional*: If you will be using replication, you must archive all of the DB2 log files.
7. Install DB2 Enterprise Server Edition Version 8.1.
8. Migrate your databases.
9. *Optional*: Migrate DB2 Explain tables.

For details on the above steps, see Part 2. ″Migrating your DB2 server″ in the *Quick Beginnings for DB2 Servers* document. This document is available from the DB2 Technical Spport Library (http://www.ibm.com/software/data/db2/library/); you will need to navgate to the DB2 Technical Support Version 8 Information Center.

## Upgrading the WebSphere Commerce development environment

Once you have backed up your WebSphere Commerce and the WebSphere Commerce development environment assets, you can upgrade the WebSphere Commerce development environment. This includes the installation of WebSphere Studio Application Developer, Version 5.1, and the installation process upgrades your system to DB2 Universal Database™ Enterprise Edition Version 8.1.

**Note:** It is important that you do not uninstall VisualAge for Java or an earlier version of WebSphere Studio Application Developer that you used for WebSphere Commerce Studio, Version 5.4, as you will need to export your assets and projects from these applications into WebSphere Studio Application Developer, Version 5.1. However, to avoid conflict, ensure that you install WebSphere Commerce - Express Developer Edition, Version 5.5 in a different directory than that of VisualAge for Java or an earlier version of WebSphere Studio Application Developer. Note also that your VisualAge for Java projects will not work on the WebSphere Studio Application Developer, Version 5.1 workspace.

The WebSphere Commerce - Express Developer Edition, Version 5.5 installation program completes the following tasks:

- Installs the WebSphere Commerce development environment code, which includes WebSphere Studio Application Developer, Version 5.1, and if you use DB2 as your database management system, the installation process upgrades your system to DB2 Universal Database Enterprise Edition Version 8.1.
- Creates a new directory structure called \WebSphere\CommerceDev55 with the following subdirectories:
  - \properties

This directory contains the properties files for the WebSphere Commerce 5.5 level of code.

– \bin

This directory contains the database migration scripts.

- Copies the original wcs_instances files into the new \WebSphere\CommerceDev55\instances directory structure and updates the files to reflect the new structure.

To install the WebSphere Commerce development environment, do the following (for detailed installation instructions, refer to *WebSphere Commerce Studio Installation Guide*):

__ 1. Insert WebSphere Commerce - Express Developer Edition CD 1 in your CD-ROM drive. If you have autoplay enabled, the installation process begins. If not, run setup.exe.

__ 2. Remove the \WebSphere\CommerceDev\bin directory path from the system environment path. To remove this directory, do the following:

   a. Access your **System Properties** window.

   b. Select **Environment Variables**. Under **System variables**, select **Path**.

   c. Edit this path and remove the old Commerce Studio Development path (for example, WebSphere\CommerceDev\bin).

   d. Click **OK**.

__ 3. Follow the prompts in the installation wizard.

__ 4. Once the WebSphere Commerce development environment installation has completed, and you have custom code created in VisualAge for Java , the custom code needs to be migrated to WebSphere Studio Application Developer, Version 5.1.

   **Note:** This step is only required if you are migrating back-end assets such as commands or data beans.

   To ease this process, start VisualAge for Java and export the following projects and groups now to JAR files as outlined in "Exporting task commands, controller commands, and data beans to a JAR file" and "Exporting enterprise beans to a JAR file" on page 24. (Details on transitioning custom code and logic are described in "Differences between VisualAge for Java and WebSphere Studio Application Developer" on page 30.)

## Exporting task commands, controller commands, and data beans to a JAR file

Start VisualAge for Java and export the VisualAge for Java project containing custom code for new or extended task commands, controller commands, or data beans. In this case, you must export the project or projects containing this code to a JAR file as follows:

1. In the VisualAge for Java Workbench window, select and right-click the project name or names, then click **Export**.
2. Select the **Jar file** radio button and click **Next**.
3. Specify the name of the JAR file.
4. Select the **.java** check box to export your Java files.
5. Fill in the other fields as required. Refer to the VisualAge for Java online help for more information on how to perform this task.

## Exporting enterprise beans to a JAR file

Start VisualAge for Java and export any custom enterprise beans as follows to a JAR file (you must export all the enterprise bean groups containing custom enterprise beans to a single JAR file):

### If you have minimal custom enterprise beans

If you have a minimal number of custom enterprise beans (that is, approximately ten or less), it is recommended that you re-create the enterprise beans using WebSphere Studio Application Developer, Version 5.1. For details on how to do this, refer to the *WebSphere Studio Application Developer, Version 5.1 Migration Guide*

### If you have many custom enterprise beans

Start VisualAge for Java and export any custom enterprise beans by using the Export Tool for Enterprise Java Beans 1.1. Export each of your enterprise bean groups containing custom enterprise beans 1.1 to a corresponding JAR file:

If during the export process, the **Target database type** selection box is disabled, this is an indication that you have lost your enterprise schema mapping.

1. In VisualAge for Java, add the **Export Tool for Enterprise Java Beans 1.1** feature to the workspace.
2. On the EJB page of the Workbench, right-click your enterprise bean group or groups and click **Export** > **EJB 1.1 JAR**.
3. Fill in the fields as necessary. Ensure that the **.java** check box is selected.
4. Ensure that you select your database.
5. Click **Finish**.

**Note:** Multiple enterprise bean groups cannot be exported to a single JAR file. If you have multiple EJB groups, do the following in VisualAge for Java 4.0:

a. Open the **Projects** tab
b. Export your custom EJB projects as repositories one-at-a-time.
c. Create a new project.

d. Go to the EJB tab and create an EJB group in this project.
e. Import your custom EJB project repositories that you exported in step 5b into this EJB group. If your project has been versioned, ensure that you select the option to create new or scratch editions.

6. Import the JAR file into WebSphere Studio Application Developer, Version 5.1.

## Applying the IBM WebSphere Commerce 5.5.0.2 fix pack

After you install the WebSphere Commerce development environment, you need to install the IBM WebSphere Commerce 5.5.0.2 fix pack. This fix pack includes fixes and enhancements for the migration process. This fix pack is available from the WebSphere Commerce Support site. Click on the link for the edition you are using and see the section Software downloads. Download this fix pack, and follow the instructions that are provided to install it. You should not proceed with the migration steps in this guide (such as migrating your instance or database) without installing this fix pack.

## Migrating the WebSphere Commerce development environment database schema

Once you have upgraded the WebSphere Commerce development environment, your next step is to migrate the WebSphere Commerce development environment database schema to WebSphere Commerce - Express Developer Edition, Version 5.5 level.

To migrate the WebSphere Commerce development environment database schema, you run a database schema migration scrip called migratedb54.bat as follows:

__ 1. Before running the database schema migration script, ensure that the JAVA_HOME environment variable in the setenv.bat file points to an existing version of IBM Developer Kit, Java Technology Edition on your machine. The setenv.bat file is normally located in the *WCDE_installdir*\Commerce\bin directory.

   For example, if IBM Developer Kit, Java Technology Edition is installed in the \eclipse\jre directory on your machine, set `JAVA_HOME` to `JAVA_HOME=`*`WCDE_installdir`*`\eclipse\jre` . By default, JAVA_HOME points to the WebSphere Studio 5.1 installation directory (for example, `JAVA_HOME=C:\WebSphere\Studio5\runtimes\base_v5\java\jre`).

__ 2. Before running the database schema migration script, ensure that you drop all customized foreign key references. This will prevent various database errors that you may encounter during migration. After the database schema migration, you should rebuild these foreign key references manually.

__ 3. To run the database migration script, see the section "Migrating your database" in the *WebSphere Commerce Migration Guide*.

**Notes:**

a. In the WebSphere Commerce development environment, the database schema migration scripts are available in the *WCDE_installdir*\commerce\bin directory.

b. You can run the premigratedb script, as described in *WebSphere Commerce Migration Guide*, you can do so from a command prompt; whereas the migratedb script should be run from a DB2 Command Line Processor window.

## Migrating the WebSphere Commerce instance

Once you have migrated the WebSphere Commerce development environment database schema to the WebSphere Commerce - Express Developer Edition, Version 5.5 level, your next step is to migrate the WebSphere Commerce instance.

During the WebSphere Commerce instance migration process, the following tasks are completed:

- Migrates the instance.xml file.
- Copies the store front or Web assets from VisualAge for Java into a WebSphere Studio Application Developer Web project. This includes any JSP files, HTML pages, or graphics you used with your store front.
- Extracts the JAR file containing the non-enterprise bean custom code into the WebSphere Studio Application Developer Java project called WebSphereCommerceServerExtensionsLogic.
- Migrates the JSP files that were copied when you run the database migration script, to conform to the JSP 1.2 specification.

**Note:** Instance migration involves automatically copying the store front or Web assets (that is, JSP files, HTML pages, or graphics) from VisualAge for Java to WebSphere Studio Application Developer. However, you should also ensure that you have accounted for all the assets created for your store front within WebSphere Commerce Studio, Version 5.4.

To migrate the instance, you run two instance migration scripts: wstudioimenv.bat and wstudioim.bat, as follows:

1. If you are migrating store front assets, start WebSphere Studio Application Developer, Version 5.1 and create a Web project for your store assets and note the name of the project. You will need to have this project created before running the wstudioim.bat instance migration script as you will need to specify the Web project name as a parameter value when running this script.

**Note:** If you only are only migrating back-end assets, then you can skip this step.

2. Ensure that all the values are properly set for your migration path within the product.xml file found in the *WC_installdir*\xml directory. The values for product.xml get filled in by the WebSphere Commerce 5.5 installation program, however, it is recommended that you check the values to ensure they are correct for your environment. Specifically, ensure that the <migrationFrom> element and all the child elements of this element contain values. If they do not, you must manually insert the values. The values can be obtained from the product.xml file used with WebSphere Commerce 5.4; refer to the <CommerceServer> tag.

   The <migrationFrom> element should be in the following format:

```
<migrationFrom>
  <edition>
    <name>name</name>
  </edition>
  <version>5</version>
  <release>rel</release>
  <modification>mod</modification>
  <fixpak>fixpak</fixpak>
  <path>path</path>
  <altpath>long_path</altpath>
</migrationFrom>
```

   where:

   *name* The edition of WebSphere Commerce from which you are migrating. The name can be any of the following:
   - **Pro** — Denotes migration from WebSphere Commerce Professional Edition
   - **Business** — Denotes migration from WebSphere Commerce Buesiness Edition

   *rel* The release of WebSphere Commerce from which you are migrating. Specify **4** for WebSphere Commerce 5.4.

   *mod* The modification level of WebSphere Commerce from which you are migrating. The modification level is always a value of **0**. For example, to migrate from WebSphere Commerce 5.4.**0**.1, the modification level is **0** in the product.xml file.

   *fixpak* The fix pack level of WebSphere Commerce from which you are migrating. For example, to migrate from WebSphere Commerce 5.4.0.**4**, the fix pack level is **4** in the product.xml file.

   *path* The installation path for the version of WebSphere Commerce from which you are migrating. This is typically the same as the *long_path* value. For example, to migrate from WebSphere Commerce 5.4, the path is *WC54_installdir*, such as *c:\WebSphere\WebSphere Commerce*.

*long_path*

The full or long installation path for the version of WebSphere Commerce from which you are migrating. This is typically the same as the *path* value. For example, to migrate from WebSphere Commerce 5.4, the path is *WC54_installdir*, such as *c:\WebSphere\WebSphere Commerce*.

For your reference, the following is an example of what your product.xml file needs to contain if you are migrating from WebSphere Commerce 5.4.0.1 Business Edition:

```
<migrationFrom>
  <edition>
    <name>Business</name>
  </edition>
  <version>5</version>
  <release>4</release>
  <modification>0</modification>
  <fixpak>1</fixpak>
  <path>c:\WebSphere\WebSphere Commerce</path>
  <altpath>c:\WebSphere\WebSphere Commerce</altpath>
</migrationFrom>
```

3. Open the *WCDE_installdir*\Commerce\bin\setenv.bat file and ensure the values for the following settings are correct:
   - WAS_HOME is set to *WAS_installdir*\runtimes\base_v5
   - JAVA_HOME is set to your \eclipse\jre directory on the WebSphere Commerce development environment machine; for example, *WCDE_installdir*\runtimes\base_v5\java\jre or C:\WebSphere\Studio5\runtimes\base_v5\java\jre.
   - WCS_HOME is set to the *WC_installdir*; that is, the home directory of where you installed WebSphere Commerce 5.5. For example, C:\Program Files\WebSphere\CommerceServer55
4. Edit the *WC_installdir*\bin\wstudioimenv.bat file and ensure the values for the following settings are correct:
   - SET WCIM_MIGRATE_FROM should be SET WCIM_MIGRATE_FROM=54
   - SET WC_PATH is set to the *WC_installdir*; that is, the directory of where you installed WebSphere Commerce 5.5. For example, C:\Program Files\WebSphere\CommerceServer55
   - SET ANT_PATH is set to *WSAD_installdir*\runtimes\base_v5\lib
   - SET WORK_DIR is set to a temporary working directory for *WCDE_installdir*; for example, *WCDE_installdir*\temp
   - SET LOG_FILE is set to the name of the file to log information about the instance migration.
   - SET INSTANCE is the name of the WebSphere Commerce 5.5 instance.
5. From a command prompt, change to the *WCDE_installdir*\bin directory and run the first script by typing:

```
wstudioimenv.bat
```

6. From a command prompt, change to the *WCDE_installdir*\bin directory and run the second script by typing the following as one line:

   ```
   wstudioim.bat WSAD_workspacedir VAJ_installdir
     custom_code_JAR_file store_properties_dir
   ```

   where:

   *WSAD_workspacedir*
   : The path for your WebSphere Studio Application Developer, Version 5.1 workspace, specified when you installed the WebSphere Commerce development environment. For example, c:\WebSphere\workspace_db2.

   *VAJ_installdir*
   : The installation path for VisualAge for Java. The default installation path is \Visual Age for Java. For example, d:\Visual Age for Java.

   *custom_code_JAR_file*
   : The JAR file containing custom code such as customized enterprise beans, commands, or data beans. For example, d:\myCustomCode.jar.

   *store_properties_dir*
   : The path for store properties files of your previous version of WebSphere Commerce. An example path is c:\WebSphere\CommerceServerDev\wc.ear\wcstores.war \WEB-INF\classes.

   **Note:** If you are only migrating store front assets, for the custom_code_JAR_file parameter, specify a blank value (that is, use two double quotation marks to indicate the blank value). For example, in this case, you can execute the instance migration script as follows:

   ```
   wstudioim.bat WSAD_workspacedir VAJ_installdir ""
   store_properties_dir
   ```

7. View the log file *WCDE_installdir*/temp/logs/wstudioim.log to see if the program ran successfully.
8. Edit the migrated instance XML file as follows:
   a. Search for WorkspacePath=""
   b. Add thethe WebSphere Commerce development environment workspace path between the quotation marks (for example, WorkspacePath="D:\WEBSPH~1\WORKSP~1").
9. Import your custom enterprise beans to the WebSphereCommerceServerExtensionsData enterprise bean project within WebSphere Studio Application Developer, Version 5.1. Start WebSphere

Studio Application Developer, Version 5.1 and import the enterprise bean JAR file or files exported from VisualAge for Java in step "Exporting enterprise beans to a JAR file" on page 24 as follows:

a. In WebSphere Studio Application Developer, from the J2EE perspective, select **File** > **Import** > **EJB JAR file**. Click **Next**.
b. Specify the JAR file containing your enterprise beans.
c. Specify the enterprise bean project where the enterprise beans will be imported by selecting the existing project called **WebsphereCommerceServerExtensionsData**.
d. Click **Next** and select all the enterprise bean files from your JAR file.
e. Click **Finish**. If you have any errors (they will be listed in the Tasks view), refer to "Troubleshooting tips for migrating enterprise beans to WebSphere Studio Application Developer" on page 35 to troubleshoot them. After you have imported your enterprise beans, follow the instructions in "Locating enterprise bean information" on page 36 to determine where to find your enterprise bean and method properties, your schemas, maps, and so on.

## Transitioning your customized or extended code

Once you have migrated the WebSphere Commerce instance, your next step is to transition your extended or modified business logic and code contained within your VisualAge for Java workspace to the WebSphere Studio Application Developer, Version 5.1 level. To help with the complex task of transitioning custom or extended code, two tutorials are provided within this guide. Refer to Part 4, "Tutorials for transitioning your code," on page 49 to learn more about migrating customized or extended code.

### Differences between VisualAge for Java and WebSphere Studio Application Developer

Before you transition code from your VisualAge for Java workspace to the WebSphere Studio Application Developer, it is important to understand the differences between the two software applications. The following highlights the main differences between VisualAge for Java and WebSphere Studio Application Developer:

- The Enterprise Java Beans (EJB) specification level has changed from 1.0 to 1.1. That is, the 1.1 specification is for the WebSphere Commerce development environment using WebSphere Studio Application Developer, Version 5.1.
- For Web applications, the JSP level changed from 1.1 to 1.2.
- For Web applications, the Servlet level remains at 2.3.
- The level of the Java 2 platform that is supported has changed from 1.2 to 1.3 (The compiler can target 1.4 code generation, but the WebSphere Application Server run-time environment is still 1.3.)

- The Visual Composition Editor has been replaced by the Visual Editor for Java.
- VisualAge for Java version control and the propriety source code repository have been replaced by support for software configuration management (SCM) plug-ins. The VisualAge for Java Tools API has been replaced by the WebSphere Studio Workbench plug-in architecture.
- The VisualAge for Java XML tools have been replaced by WebSphere Studio Application Developer XML tools.
- The VisualAge for Java project concept has been replaced by multiple types of WebSphere Studio Application Developer projects.
- Convertors in VisualAge for Java EJB AccessBeans (not Data AccessBeans) are not available in WebSphere Studio Application Developer EJB AccessBeans. Use enterprise bean convertors and composers in the underlying enterprise bean instead. For more information about enterprise bean convertors and composers, refer to the online help documentation.

## Transitioning extended or modified WebSphere Commerce business logic and code

During the migration from WebSphere Commerce Studio, Version 5.4 to WebSphere Commerce - Express Developer Edition, Version 5.5, a number of steps related to your VisualAge for Java environment are performed. These steps are as followings:

- Exporting your Java files and project resource files from VisualAge for Java.
- Starting WebSphere Studio Application Developer and creating new projects to contain your code.
- Importing your Java and project resource files into WebSphere Studio Application Developer.
- Migrating your enterprise beans into WebSphere Studio Application Developer.
- Migrating your customized WebSphere Commerce Server enterprise beans into WebSphere Studio Application Developer.
- Setting up your test environment and testing your migrated application.

**Notes:**

1. For WebSphere Commerce 5.4, the Common Connector Framework was used. WebSphere Commerce 5.5, however, has changed to use the WebSphere Application Command Framework. Thus, when migrating custom or extended code, it is important to note the following:
   - All the customer commands that extend from WebSphere Commerce 5.4 controller commands or task command are required to be recompiled in order to pick up the new WebSphere Application Server, Version 5 runtime libraries.

- The callers of the controller commands and task commands are required to have the command context set.

2. For information on supported, deprecated, reserved, and withdrawn API, refer to the WebSphere Commerce 5.5 Development online help.

### Exporting your Java files and project resource files from VisualAge for Java

Your first step in transitioning code to the WebSphere Studio Application Developer, Version 5.1 level involves exporting Java files and project resource files from VisualAge for Java.

**Notes:**

1. There is no support for the bulk migration of versioned projects and resources from the VisualAge for Java repository. You can, however, migrate projects and resources that are in your VisualAge for Java workspace. If you want to migrate a versioned copy of a project or resource into WebSphere Studio Application Developer, you must bring it into your VisualAge for Java workspace and then migrate it.
2. If your project contains more than one kind of data (for example, enterprise beans and Java source code files), you should split up your data into different JARs based on their type.

To export your projects to a JAR file, follow these steps:

1. If the projects that you want to export are not currently in your VisualAge for Java workspace, add them to the workspace.
2. In the VisualAge for Java Workbench window, select you project or projects, right-click, and click **Export**.
3. Select the **Jar file** radio button and click **Next**.
4. Type the name of the JAR file.
5. Select the **.java** check box to export your Java files and the **resources** check box to export your resource files.
6. Fill in the other fields as required. Refer to the VisualAge for Java online help for more information on how to complete the fields.

### Starting WebSphere Studio Application Developer and creating new projects to contain your code

After you have exported your Java files and project resource files from VisualAge for Java, the next step in transitioning your code WebSphere Studio Application Developer is to start WebSphere Studio Application Developer and create appropriate projects. The following is a set of general migration guidelines to help you decide the kind of WebSphere Studio Application Developer project into which you should import your files. It is strongly recommended that you read the WebSphere Studio Application Developer

online help and become familiar with the different kinds of WebSphere Studio Application Developer projects before you create any projects or import any code.

- If your code is part of a Web application, you should import the code into a Web project as follows:
  - Import all Java files, such as a controller command or a task command, into the Web project source directory (the proper hierarchy based on their package statements will automatically be created by WebSphere Studio Application Developer).
  - Import all resource files, such as JSP pages, into the Web project Web content directory.
- If your code is straight Java, you should import the code into a Java project.
- If your code is enterprise beans, you should import the code into an enterprise bean project.

### Importing your Java and project resource files into WebSphere Studio Application Developer

After starting WebSphere Studio Application Developer and creating new projects to contain your code, the next step in transitioning your code to WebSphere Studio Application Developer is to import your Java and project resource files into WebSphere Studio Application Developer.

**Note:** When you import your files into WebSphere Studio Application Developer, ensure that they are imported to the appropriate directory. It is recommended that you read the WebSphere Studio Application Developer online help and become familiar with the different kinds of WebSphere Studio Application Developer projects before importing your code. This will help you determine which folders should contain which kind of code.

To import your Java and project resource files into WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Select **File** > **Import** > **Zip file**. Click **Next**.
3. Browse to the appropriate JAR file.
4. Select the files you want to import and the project or folder you want to contain your files.

### Migrating your enterprise beans into WebSphere Studio Application Developer

After importing your Java and project resource files into WebSphere Studio Application Developer, the next step in transitioning your code to WebSphere Studio Application Developer is to migrate your enterprise beans into

WebSphere Studio Application Developer. This task involves three subtasks: exporting the enterprise beans from VisualAge for Java, importing the enterprise beans into WebSphere Studio Application Developer, and generating deploy code.

**Exporting your enterprise beans from VisualAge for Java:**

1. In VisualAge for Java, add the **Export Tool for Enterprise Java Beans 1.1** feature to the workspace if you have not already added it.
2. On the EJB page of the Workbench, right-click on your enterprise bean group or groups and click **Export** > **EJB 1.1 JAR**.
3. Fill in the fields as necessary. Ensure that the **.java** check box is selected.
4. Ensure that you select your database.
5. Click **Finish**.

**Importing your enterprise beans into WebSphere Studio Application Developer:**

1. In WebSphere Studio Application Developer, create a new EJB project and a new enterprise applications archives project. You will automatically be switched to the J2EE perspective.
2. Select **File** > **Import** > **EJB JAR file**. Click **Next**.
3. Select your JAR file, your EJB project, and your EAR project.
4. Click **Next** and select all the EJB files from your JAR file.
5. Click **Next** and select all the dependent projects. If you are unsure as to which projects you have a dependency on, select all of them.
6. Click **Finish**. If you have any errors (they will be listed in the Tasks view), refer to "Troubleshooting tips for migrating enterprise beans to WebSphere Studio Application Developer" on page 35 to troubleshoot them. After you have imported your enterprise beans, follow the instructions in "Locating enterprise bean information" on page 36 to determine where to find your enterprise bean and method properties, your schemas, maps, and so on.

**Editing methods to avoid java.rmi.RemoteException errors:**

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the Project Navigator view, open WebSphereCommerceServerExtensionData\ejbModule\com.ibm.commerce.sample.objects*enterprise_bean*

   where *enterprise_bean* is the name of the enterprise bean you imported into WebSphere Studio Application Developer; for example, MyEnterpriseBean.java.
3. Modify the following methods so that they do not throw java.rmi.RemoteException errors:

```
ejbLoad()
ejbStore()
unsetEntityContext()
```

For example, change `public void ejbLoad() throws java.rmi.RemoteException` to `public void ejbLoad()`

4. Save your changes.

**Changing the access isolation level:**

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the Project Navigator view, open ejb-jar.xml under \WebSphereCommerceServerExtensionData\ejbModule\META-INF
3. Click **Access**.
4. Select the node **Committed**, which is below **Isolation Level**, and click**Remove**.
5. Click **Add**, select **Repeatable read**, and click **Next**.
6. Select the name of the enterprise bean and click **Next**.
7. Expand name of the enterprise bean and select the check box for all home methods (that is, the first icon under name of the enterprise bean).
8. Click **Finish**.
9. Verify that the node **Repeat** now appears under **Isolation Level**.
10. Save your changes.

**Generating deploy code:**

1. In WebSphere Studio Application Developer, go to the J2EE Hierarchy view, expand the EJB Modules folder, and select the newly imported EJB JAR file.
2. From the **EJB JAR** pop-up menu, click **Generate** > **Deploy and RMIC Code**. Select to generate code for all your enterprise beans.

*Troubleshooting tips for migrating enterprise beans to WebSphere Studio Application Developer:* The following are some troubleshooting tips that may help when migrating enterprise beans to WebSphere Studio Application Developer:

- When migrating the method finder helpers, the finder helper interfaces will disappear, as they have moved into an XML description file. This will generate a problem since your finder helper object usually implements this interface; the implementation must be removed.
- If you use inheritance in your enterprise beans, and you have built your own custom finders, they may break, since the mapping of fields in the generated code is different in WebSphere Studio Application Developer. To fix this, go to the `EJSCMPxxxxxx` generated class, find the select statement generated for the `findbyPrimaryKey`, and use it as a template.

- If you set up the Preferences page within WebSphere Studio Application Developer, to stop an automatic build from being done every time you save changes, you must perform the following steps to generate the enterprise bean deployed code and RMIC stubs:
  1. Select your enterprise bean project, right-click and select **Generate** > **Deploy and RMIC code**.
  2. Since the RMIC compiler can only see compiled code and since the generated Java classes were not compiled yet, you will receive an error. After you receive this error, switch to Java perspective and build the project.
  3. Repeat step 1. You should not receive any errors this time.
  4. Repeat step 2 to compile your newly generated stubs so you do not receive run-time errors when you deploy your enterprise beans.

*Locating enterprise bean information:* The following table contains a list of enterprise bean items and where to find your enterprise beans after you have migrated them to WebSphere Studio Application Developer:

*Table 4. Locating enterprise bean information after migrating to WebSphere Studio Application Developer, Version 5.1*

| Item | Location |
|---|---|
| Enterprise beans (fields, classes) | Expand the EJB Modules folder in the J2EE Hierarchy view. |
| Enterprise beans (finder classes, access beans, generated classes) | Expand the EJB Modules folder in the Navigator view and go to the `com` directory (or your package structure). |
| Associated information | Expand the EJB Modules folder in the J2EE Hierarchy view and select the enterprise bean JAR file you want to work with. From its pop-up menu, select **Open With** > **EJB Deployment Descriptor**. In the Deployment Descriptor, select the **Overview** tab. |
| Finder helper description | Expand the EJB Modules folder in the J2EE Hierarchy view and select the enterprise bean JAR file you want to work with. From its pop-up menu, select **Open With** > **EJB Deployment Descriptor**. In the Deployment Descriptor, select the **Beans** tab. |
| Mapping or schema description | Expand the EJB Modules folder in the J2EE Hierarchy view and select the EJB JAR file you want to work with. From its pop-up menu, select **Generate** > **EJB to RDB mapping**. |

*Table 4. Locating enterprise bean information after migrating to WebSphere Studio Application Developer, Version 5.1 (continued)*

| Item | Location |
| --- | --- |
| Transaction demarcation | Expand the EJB Modules folder in the J2EE Hierarchy view and select the enterprise bean JAR file you want to work with. From its pop-up menu, select **Open With** > **EJB Deployment Descriptor Editor**. In the EJB Deployment Descriptor Editor, select the **Beans** tab. |
| Isolation levels or find for update or read only methods marking | Expand the EJB Modules folder in the J2EE Hierarchy view and select the EJB JAR file you want to work with. From its pop-up menu, select **Open With** > **EJB Deployment Descriptor**. In the EJB Deployment Descriptor, select the **Access** tab. |
| EJB environment variables | Expand the `EJB Modules` folder in the J2EE Hierarchy view and select the enterprise bean JAR file you want to work with. From its pop-up menu, select **Open With** > **EJB Deployment Descriptor**. In the EJB Deployment Descriptor , select the **Beans** tab. |

*Migrating EJB access beans:* When enterprise beans are exported from VisualAge for Java Enterprise Edition, Version 4.0 using the Export Tool for Enterprise Java Beans 1.1, the metadata for any associated Java bean wrapper and copy-helper access beans will also be exported. Rowset access beans are not supported by WebSphere Studio Application Developer, so the metadata for these access beans will not be exported.

*Migrating customer finder helpers:* When you export enterprise beans from VisualAge for Java Enterprise Edition, Version 4.0 using the Export Tool for Enterprise JavaBeans™ 1.1, or when 1.0 JAR files are deployed with the Deployment Tool for Enterprise JavaBeans (that is, the EJB Deploy Tool), the finder helper interfaces are migrated to the extension document. If the JAR file is an enterprise bean JAR file, the metadata is also migrated to the extension document from the finder helper interfaces. However, migration of the finder helper interfaces to the extension document only occurs if the finder descriptors in the JAR file are not found in the extension document. If the enterprise beans are exported using the Export Tool for Enterprise Java Beans 1.1, the redundant classes are filtered from the exported JAR. If the enterprise

beans are not exported using the Export Tool for Enterprise Java Beans 1.1 and are imported along with the redundant classes, the classes are simply ignored.

### Migrating your customized WebSphere Commerce Server enterprise beans into WebSphere Studio Application Developer

After migrating your enterprise beans into WebSphere Studio Application Developer, the next step in transitioning your code to WebSphere Studio Application Developer is to migrate your customized WebSphere Commerce Server enterprise beans into WebSphere Studio Application Developer. This task involves two subtasks: migrating code and meta-data changes into existing WebSphere Commerce Server projects in WebSphere Studio Application Developer, and generating deploy code and the access bean.

**Migrating code and meta-data changes into existing WebSphere Commerce Server projects in WebSphere Studio Application Developer:** To migrate code and meta-data changes into existing WebSphere Commerce Server projects in WebSphere Studio Application Developer, do the following:

1. In WebSphere Studio Application Developer, select the WebSphere Commerce Server enterprise bean project to which you want to make changes.
2. Merge your Java code changes. To do this, refer to the WebSphere Studio Application Developer documentation on how to make Java code changes in an enterprise bean project.
3. Merge your table or column definitions. To do this, refer to the WebSphere Studio Application Developer documentation on how to modify existing database table definitions in an enterprise bean project.
4. Merge your container managed persistence (CMP) fields. Refer to the WebSphere Studio Application Developer documentation on how to add or modify CMP fields for an enterprise bean.
5. Merge finder methods. Refer to the WebSphere Studio Application Developer documentation on how to add or modify finder methods in the deployment descriptor for an enterprise bean project.
6. Repeat the above steps for each WebSphere Commerce Server enterprise bean project you want to change.

**Note:** You may encounter warnings or errors that during the code merge process. This is normal and any errors are fixed when you generate the deploy code and access bean, as described in "Generating deploy code" on page 63.

### Generating deploy code

To regenerate the deployed code, do the following:

1. In WebSphere Studio Application Developer, go to the the J2EE perspective, and from the J2EE hierarchy, expand **EJB Modules**.
2. Right-click the newly-created EJB JAR file and select **Generate** > **Deploy and RMIC Code**.
3. Select **All EJBs** and click **Finish**.

**Note:** In addition, refer to the following information about migrating enterprise beans into WebSphere Studio Application Developer:

- "Troubleshooting tips for migrating enterprise beans to WebSphere Studio Application Developer" on page 35
- "Locating enterprise bean information" on page 36
- "Migrating EJB access beans" on page 37
- "Migrating customer finder helpers" on page 37

### Setting up your test environment and testing your migrated application

After migrating your enterprise bean files into WebSphere Studio Application Developer, the last step in transitioning your code to WebSphere Studio Application Developer is to set up your test environment and test your migrated application.

To start the server instance, follow these steps:

1. In WebSphere Studio Application Developer, ensure that you close the local Web server or WebSphere Application Server server. Otherwise, the WebSphere Commerce will not start successfully.
2. Check the data source property page and change the default password to your password for the data source setting you used. By default, the **Default password** field is blank.
3. Provide the data source setting and ensure that you set the correct URL for the data source. If you do not ensure that the URL is correct, you will encounter errors when starting the Commerce Server and you will not be able to create a data source successfully. For DB2, the default JDBC data source URL is `jdbc:db2:database_name`; for example, `jdbc:db2:demo_dev`. Ensure that the data source URL is such.
4. Update the the WebSphere Commerce Server data source with the correct database user password as follows:
   a. Start WebSphere Studio Application Developer by selecting **Start** > **Programs** > **Start WebSphere Studio** > **Application Developer 5.0**.
   b. In the J2EE Hierarchy view of the J2EE Perspective, expand **Servers** and double-click **WebSphereCommerceServer**. The WebSphereCommerceServer view displays.
   c. In the WebSphereCommerceServer view, select the **Data sources** page.
   d. On the Data sources page, expand **Server Settings**.

e. In the **JDBC provider list** table, select **DB2 JDBC Driver**.

   After selecting a JDBC provider, the tables below the JDBC provider list table updates.

f. In the **Data source defined in the JDBC provider selected above** table, select **jdbc/WebSphere Commerce DB2 DataSource** *instance_name*.

   where *instance_name* is the name of the WebSphere Commerce Studio instance. The default instance name is Demo_Dev.

g. Click **Edit**. The Modify Data Source wizard opens.

h. Ensure that the data source JDNI name is as follows:

   `jdbc/WebSphere Commerce DB2 DataSource instance_name`

   For example, if your WebSphere Commerce instance name is called **demo**, the JDNI name should be jdbc/WebSphere Commerce DB2 DataSource demo. If the data source JDNI name value is not as such, the WebSphere Commerce Server will not start. In this case, you will need to modify the data source JDNI name to match the value within the instance.xml file.

i. In the Modify Data Source wizard, replace any value in the **Default user password** field with the password for the ID shown in the **Default user ID** field.

j. Click **Finish** to update the information and close the Modify Data Source wizard.

k. Save the updates to the configuration by selecting **File** > **Save WebSphereCommerceServer**.

5. Go to the Servers view. If you cannot find it in the perspective, on the main menu select **Perspective** > **Show View** > **Servers**.
6. Right-click the server instance and select **Publish**.
7. When the publishing is complete, close the pop-up menu by clicking **OK**.
8. Right-click the server instance and select **Start**. The Console view displays.
9. Follow the startup process for the instance.

You are now ready to test your migrated code by publishing a store. Follow these steps to access and setup the store using the WebSphere Commerce Administration Console:

1. Open a browser window and type the following URL: https://*host_name*/webapp/wcs/admin/servlet/ToolsLogon?XMLFile=adminconsole.AdminConsoleLogonhttps
2. On the Administration Console Logon page, type `wcsadmin` as the user name, provide your password, and click **Logon**. (By default, the user name and password are both set to `wcsadmin` during installation, but you must change the password the first time you use the WebSphere Commerce Administration Console.)

3. After you log into the WebSphere Commerce Administration Console, select **Site**.
4. From the **Store** within the WebSphere Commerce Administration Console main page, select **Publish**.
5. Complete the information within this wizard, clicking **Next** to proceed through the pages.
6. On the Options page, click **Finish** to publish your store.

Note that in comparison to the WebSphere Test Environment in VisualAge for Java, you are able to perform HTTPS requests due to the set up you have just completed.

## Migrating the payment instance and database

Once you have transitioned your custom code to WebSphere Studio Application Developer, Version 5.1, your next step is to migrate the WebSphere Commerce payment instance and database to be used with WebSphere Commerce Payments (formerly Payment Manager). If you use an external payment system, then you do not have to complete this section, but you must ensure that your payment system works properly with WebSphere Commerce 5.5 and the WebSphere Commerce development environment.

For WebSphere Commerce 5.4, the payment database was required to be installed on a different machine than the one containing WebSphere Commerce. For WebSphere Commerce 5.5, you now have the option of installing the payment database locally on the same machine as the WebSphere Commerce development environment. Thus, for version 5.5, two scenarios for payments is possible:

- If you keep payments remote, refer to the *WebSphere Commerce Migration Guide* for details on how to migrate a Commerce Payments instance and database using the WebSphere Commerce Instance Migrator tool .
- If you want to install the payment database on the same machine that contains the WebSphere Commerce development environment, you must migrate your payment database and ensure that the payment instances created in WebSphere Studio Application Developer, Version 5.1 use the migrated payment database.

# Part 3. Post-migration

This part describes step you should follow after you have migrated the WebSphere Commerce development environment.

# Chapter 4. Post-migration actions

After you have completed the migration steps outlined in Part 2, "Migration flow," on page 19, and you are migrating from WebSphere Commerce Studio, Version 5.4, ensure that you have completed the following:

- The JavaServer Page templates in your store comply to the JavaServer Page 1.2 specification, created by Sun Microsystems. The JSP templates used in stores running in WebSphere Commerce 5.4 were required to support the JSP 1.1 specification. When migrating your store toWebSphere Commerce 5.5, you must ensure that the JSP templates in your store comply to the JavaServer Page 1.2 specification created by Sun Microsystems. For information about the JavaServer Page 1.2 specification, refer to Sun Microsystem's Java Web site at www.java.sun.com.
- The JavaServer Pages Specification 1.2 states that the only language supported is ″java″. Thus, the following page language declaration in your JSPs are no longer valid:

  ```
  <%@ page language="JAVA" %>
  ```

  Note that the WebSphere Commerce Instance Migrator tool (used for payment migration) will convert all occurrences of <%@ page language=″JAVA″ %> to <%@ page language=″java″ %> on your behalf.
- If you are using AbstractAccessBean.getInitContext() to lookup the initial context in your JSPs, it is recommended that you change this to AbstractAccessBean.getInitContext(null,null).
- Remove the `<jsp:root>` and `</jsp:root>` sections in your JSP files since they are only valid if your JSP files are XML documents. If your JSP files are not XML documents (that is they are in JavaServer Pages format), you should remove the `<jsp:root>...</jsp:root>` sections from these JSP files. For more information on this and XML Documents, refer to section ″JSP.5.2 JSP Documents″ of the *JavaServer Pages Specification (Version 1.2)* available from Sun Microsystems and the section entitled ″Updating the WebSphere Commerce 5.4 JSP files″ within the *WebSphere Commerce Migration Guide*.
- Import the java.util.* packages. In WebSphere Application Server, Version 5, JSP files that use the ″Vector″ directive have to explicitly include java.util.Vector package. If you have the following line in your JSPs, your JSPs will not require any changes to work in WebSphere Commerce 5.5:

  ```
  <%@ page import="java.util.*" %>
  ```

For example, if you have migrated the ToolTech sample store, in order the the JSP pages to display properly, you need to add the following line to the top of the JSP page (for example to the top of the CatalogItemAdd.jsp page):

```
<@ page import="java.util.Vector"%>
```

If you do not import the java.util package and you use classes inside of the package, you will need to make the following changes to your JSP files. Common classes used in WebSphere Commerce are as follows:

**Enumeration**
To import the specific class, use:

```
<%@ page import="java.util.Enumeration" %>
```

**Vector** To import the specific class, use:

```
<%@ page import="java.util.Vector" %>
```

**ResourceBundle**
To import the specific class, use:

```
<%@ page import="java.util.ResourceBundle" %>
```

For more information on classes provided with WebSphere Commerce 5.5, see the WebSphere Commerce Production and Development online help.

## Environment specific post-migration actions

Some of the post-migration actions depend upon your environment, as described here.

## Environment B

For Environment B, where the development environment and WebSphere Commerce coexist but do not share assets, do the following:

1. Rename the *instance*.xml file to a unique name (for identification purposes only). For example, rename it from demo.xml to demo2.xml.
2. Modify the wcs_instances file in the WebSphere Commerce instances directory (WebSphere\WCS\instances) to point to the new XML file.
3. Edit the *instance*.xml file located in the CommerceServerDev\instances*instance*\xml directory and update the `name` entry for the database to the new name (for example, `Mall2`). Following is an example:

```
<Database>
<DB DBMSName="DB2"
    RunDB2SG="true"
    SummaryTable="true"
    OraUserID=""
    DBAName=""
```

```
    RemoteDB="false"
    DBAPwd=""
    DBServerPort=""
    DBNode=""
    DBHost=""
    DBUserID=""
    DBUserPwd=""
    name="Mall2"
    active="true"
    StagingEnable="false" />
</Database>
```

# Part 4. Tutorials for transitioning your code

The following tutorials illustrate how to effectively migrate customized or extended code. Each tutorial is an extension of the tutorials within the *WebSphere Commerce Programming Guide and Tutorials*; for example, the programming guide tutorial illustrates how to extend a controller command, and the migration tutorial explains how to migrate the extended command to WebSphere Studio Application Developer.

Before you try out the tutorials, ensure that you have migrated all the VisualAge for Java store assets into your WebSphere Studio Application Developer development environment. The following is a list of steps to migrate your JSP files into WebSphere Studio Application Developer:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Click **File** > **Import** > **Folder**, and click **Next**.
3. Browse to the directory *VAJ_installdir*\ide\project_resources\IBM WebSphere Test Environment\hosts\default_host\default_app\web
4. Select all files to import, including sub-folders.
5. Choose the folder \Stores\Web Content to contain your files.

# Chapter 5. Migration flow for the tutorials

The following is a high level view of the end-to-end scenario or flow for the tutorials outlined in this guide, using the scenario of three machines: Machine A runs WebSphere Commerce 5.4 for production purposes, Machine B is an empty machine and is for migration, and finally, Machine C is the development machine running VisualAge for Java. The following are the steps required for this scenario:

1. Install required software stack onto Machine B.
2. Run an instance migration from Machine A to B. This may be a remote instance migration.
3. Copy the database from Machine A to Machine B.
4. Run database migration on Machine B.
5. Install WebSphere Studio Application Developer, Version 5.1 on Machine C.
6. Export and import custom code and enterprise beans from VisualAge for Java to WebSphere Studio Application Developer
7. Migrate and unit test the custom code and enterprise beans.
8. Deploy (assemble) custom code and enterprise beans into the migrated EAR file resulted from step running the instance migration from Machine A to B. For details on deployment, refer to the *WebSphere Commerce Programming Guide and Tutorials* guide.
9. Deploy (install) the EAR. For details on deployment, refer to the *WebSphere Commerce Programming Guide and Tutorials* guide.
10. Test that all custom code and enterprise beans works appropriately.
11. Shut down production on Machine A.
12. Copy the database from Machine A to Machine B.
13. Run database migration on Machine B.
14. Switch configuration to use Machine B as the production server.
15. Start WebSphere Commerce 5.5 on Machine B.

# Chapter 6. Tutorial A: Migrating an extended or customized controller command

The following tutorial walks you through migrating an extended controller command. This migration involves several sub-steps: migrating the controller command itself, migrating the JSP file displaying the command, migrating the enterprise bean associated with the command, and testing that everything migrated successfully. The tutorial within the *WebSphere Commerce Programmer's Guide* for WebSphere Commerce 5.4 illustrated how to extend the existing OrderProcess controller command so that the total bonus points that have been accumulated on the purchase are displayed on the order confirmation page. For this tutorial, you will the newly extended command, OrderProcessCmdBonusImpl, as a model for the controller command migration instructions.

## Migrating an enterprise bean

In the same *WebSphere Commerce Programming Guide and Tutorials* guide tutorial, you created a BonusBean enterprise bean. Migrating this enterprise bean to WebSphere Studio Application Developer involves several sub-steps: exporting the enterprise bean as a 1.1. JAR, deleting the ExtensionJDBCHelperBean in the WebSphereCommerceServerExtensionData folder, importing the enterprise bean into WebSphere Studio Application Developer, and generating the deploy code.

### Exporting your enterprise bean from VisualAge for Java

1. In VisualAge for Java, add the **Export Tool for Enterprise Java Beans 1.1** feature to the workspace if you have not already added it.
2. On the EJB page of the Workbench, right-click the **WCSSamplesEntity Bean** enterprise bean group and click **Export** > **EJB 1.1 JAR**.
3. Fill in the fields as necessary. Ensure that the **.java** check box is selected.
4. Select your database as UDB DB2.
5. Ensure that you select your database.
6. Click **Finish**.

### Deleting the ExtensionJDBCHelperBean in the WebSphereCommerceServerExtensionData folder

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the J2EE Hierarchy view and expand **EJB Modules**.
3. Expand the **WebSphereCommerceServerExtensionsData** folder.
4. Expand **Session Beans**.

5. Right-click **ExtensionJDBCHelperBean** and click **Delete**.
6. Click **Select All** and click **OK**.

### Importing your enterprise bean into WebSphere Studio Application Developer

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. Select **File** > **Import** > **EJB JAR file**. Click **Next**.
3. Select your the EJB project called **WebSphereCommerceServerExtensionsData**.
4. Select the EAR project called **WebSphereCommerceServer**.
5. Click **Next** and select all the EJB files from your JAR file.
6. Click **Next** and select all the dependent projects. If you are unsure as to which projects you have a dependency on, select all of them.
7. Click **Finish**.

### Editing methods to avoid java.rmi.RemoteException errors

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the Project Navigator view, open **WebSphereCommerceServerExtensionData\ejbModule \com.ibm.commerce.sample.objects\BonusBean.java**
3. Modify the following methods so that they do not throw java.rmi.RemoteException errors:

   ```
   ejbLoad()
   ejbStore()
   unsetEntityContext()
   ```

   For example, change `public void ejbLoad() throws java.rmi.RemoteException` to `public void ejbLoad()`
4. Save your changes.

### Changing the access isolation level

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the Project Navigator view, open ejb-jar.xml under \WebSphereCommerceServerExtensionData\ejbModule\META-INF
3. Click **Access**.
4. Select the node **Committed**, which is below **Isolation Level**, and click**Remove**.
5. Click **Add**, select **Repeatable read**, and click **Next**.
6. Select **Bonus** and click **Next**.
7. Expand **Bonus** and select the check box for all home methods (that is, the first icon under **Bonus**).
8. Click **Finish**.

9. Verify that the node **Repeat** now appears under **Isolation Level**.
10. Save your changes.

### Generating deploy code

Since you have modified code, you must regenerate its deployed code.

To regenerate the deployed code, do the following:

1. In WebSphere Studio Application Developer, go to the the J2EE Hierarchy view.
2. Expand the **EJB Modules** folder.
3. Right-click **WebSphereCommerceServerExtensionsData** and select **Generate** > **Deploy and RMIC Code**.
4. Select **All EJBs** and click **Finish**.

## Migrating a customized JSP file

In the same *WebSphere Commerce Programming Guide and Tutorials* guide tutorial, you modified the confirmation.jsp template to display new business logic that you added to the order process business logic. To migrate a customized JSP file to WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Click **File** > **Import** > **Folder** and click **Next**.
3. Browse to the directory *VAJ_installdir*\ide\project_resources\IBM WebSphere Test Environment\hosts\default_host\default_app\web\store_directory
4. Select confirmation.jsp to import.
5. Choose the folder \Stores\Web Content to contain your files.

## Migrating an extended controller command

To migrate an extended controller command, you must first export the project to a JAR file, import the Java and resource files into WebSphere Studio Application Developer, and then compile the migrated code.

To export your `_WCSamples` project to a JAR file, follow these steps:

1. In the VisualAge for Java Workbench window, select **_WCSamples project**, right-click, and click **Export**.
2. Select the **Jar file** radio button and click **Next**.
3. Specify the name of the JAR file.
4. Select the **.java** check box to export your Java files.

5. Fill in the other fields as required. Refer to the VisualAge for Java online help for more information on how to perform this task.

To import your Java and resource files into WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Select **File** > **Import** > **Zip file**. Click **Next**.
3. Browse to the appropriate JAR file created above.
4. Select all .java files in the JAR to import.
5. Choose the folder \WebSphereCommerceServerExtensionsLogic\src to contain your files.

To compile migrated code in the Web project, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the J2EE perspective.
2. From the Project Navigator view, right-click **WebSphereCommerceExtensionsLogic**.
3. Select **Properties**.
4. Select **Java Build Path**.
5. Select the **Projects** tab.
6. Select **WebSphereCommerceExtensionsData**.
7. Select the **WebSphereCommerceServerExtensionsLogic** project.
8. Right-click and select **Build Project**.

## Testing your new logic in the Express Store sample store

Before you can verify your new business logic, you must start your server. To do this, from the Servers view, right-click **WebSphereCommerceServer** and select **Start**.

To verify your new business logic in Express Store that is running in WebSphere Studio Application Developer, do the following:

1. Once the WebSphere Commerce Server instance has started inside of WebSphere Studio Application Developer, open a browser and type the URL for your store's home page. For example, type the following URL:

   http://hostname/webapp/wcs/stores/servlet/StoreCatalogDisplay?storeId=*store_Id*&catalogId=*catalog_Id*&langId=-1

   where *store_Id* is the identifier for your store and *catalog_Id* is the identifier for your store's catalog.
2. Select and purchase a product.

3. After you have purchased a product, the order confirmation displays the number of bonus points that were earned on the order.

# Chapter 7. Tutorial B: Migrating an extended or customized entity bean and task command

The following tutorial walks you through migrating an extended entity bean and an extended task command. This migration involves several sub-steps: migrating the task command , the extended data bean, the enterprise bean associated with the command, the JSP file displaying the command, as well as testing that everything migrated successfully. The second half of the tutorial in the *WebSphere Commerce Programmer's Guide* for WebSphere Commerce 5.4 illustrated how to customize an existing task command, GetProductContractUnitPriceCmd, and its corresponding interface. As part of this tutorial, you will create new business logic to calculate a discounted price, based upon the customer's balance of bonus points. This calculation is performed by a new task command. The newly extended command, GetNewProductContractUnitPriceCmd, and its interface, GetNewProductContractUnitPriceCmdImpl, will be used as a model for this migration tutorial.

## Migrating an extended task command

To migrate an extended task command, you must first export the project to a JAR file, import the Java and resource files into WebSphere Studio Application Developer, and then compile the migrated code.

To export your `_WCSamples` project to a JAR file, follow these steps:

1. In the VisualAge for Java Workbench window, select **_WCSamples project**, right-click, and click **Export**.
2. Select the **Jar file** radio button and click **Next**.
3. Specify the name of the JAR file.
4. Select the **.java** check box to export your Java files.
5. Fill in the other fields as required. Refer to the VisualAge for Java online help for more information on how to perform this task.

To import your Java and resource files into WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Select **File** > **Import** > **Zip file**. Click **Next**.
3. Browse to the appropriate JAR file created above.
4. Select all .java files in the JAR to import.

5. Choose the folder \WebSphereCommerceServerExtensionsLogic\src to contain your files.

To compile migrated code in the Web project, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the J2EE perspective.
2. From the Project Navigator view, open **WebSphereCommerceServerExtensionsLogic\src\com\ibm\commerce\sample\commands\GetNewBaseUnitPriceCmdImpl.java**.
3. Search for `new BigDecimal(dblBonusPrice)` and change it to `new java.math.BigDecimal(dblBonusPrice)`.
4. Save the file.
5. Select the **WebSphereCommerceServerExtensionsLogic** project.
6. Right-click and select **Build Project**.

## Migrating an extended data bean

The next step in the tutorial is to migrate the extended data bean, NewProductDataBean.java. To migrate an extended data bean, you must first export the project to a JAR file, import the Java and resource files into WebSphere Studio Application Developer, and then compile the migrated code.

To export your `_WCSamples` project to a JAR file, follow these steps:

1. In the VisualAge for Java Workbench window, select **_WCSamples project**, right-click, and click **Export**.
2. Select the **Jar file** radio button and click **Next**.
3. Specify the name of the JAR file.
4. Select the **.java** check box to export your Java files.
5. Fill in the other fields as required. Refer to the VisualAge for Java online help for more information on how to perform this task.

To import your Java and resource files into WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Select **File** > **Import** > **Zip file**. Click **Next**.
3. Browse to the appropriate JAR file created above.
4. Select all .java files in the JAR to import.
5. Choose the folder \WebSphereCommerceServerExtensionsLogic\src to contain your files.

To compile migrated code in the Web project, follow these steps:

1. Select the **WebSphereCommerceServerExtensionsLogic** project.
2. Right-click and select **Build Project**.

## Migrating an enterprise bean

In the same *WebSphere Commerce Programming Guide and Tutorials* guide tutorial, the User enterprise bean is customized in a manner such that it appears to applications that the BONUSPOINT column of the BONUS table is actually a column in the USERS table. When a new record is created in the USERS table, a corresponding record is automatically inserted into the BONUS table. In order to create this table join, a new container managed persistence (CMP) field must be added to the User entity bean. This CMP field maps to the BONUSPOINT column in the BONUS table using the Secondary Table Map feature. Migrating this enterprise bean to WebSphere Studio Application Developer involves several sub-steps: migrating the CMP filed to the User entity bean, updating the USER database schema and table mapping to include the BONUS table, and generating the deploy code and access bean for the User entity bean:

### Migrating the bonusPoint field to the User entity

In this section, you will use the enterprise bean tools to migrate the CMP field to the entity bean. The field is called bonusPoint and is eventually be mapped to the BONUSPOINT column of the BONUS table. To re-create the CMP field to the User entity bean, do the following:

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the J2EE hierarchy, expand **EJB Modules**.
3. Expand **Member-MemberManagementData EJB module** >**Entity Beans** > **Member**.
4. Right-click **User** and select **Open With** > **Deployment Descriptor Editor**.
5. Go to the Bean page and select **User EJB**.
6. On the right, click **Add** to bring up the CMP attribute wizard and provide the following information:
   - **Name**: bonusPoint
   - **Type**: int
   - **Access with getter and setter methods**: enabled
   - **Promote getter and setter methods to remote interface**: disabled
7. Click **OK**.
8. Save your EJB Deployment Descriptor changes (you can press **Ctrl+S** on your keyboard).

The bonusPoint field is displayed in the Attribute window. Note that warnings are displayed, but these are fixed when you regenerate the entity bean's code, as described in "Generating deploy code" on page 38.

### Updating the schema and table mapping to include the BONUS table

You can now update the User schema with the BONUS table, create the foreign key relationship for the new table, and create a table map between the fields of the User entity bean and the columns of the BONUS table.

To create the table schema, do the following:

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the J2EE Hierarchy view, expand on **Databases**.
3. Expand **User database schema**, and then **NULLID**.
4. Right-click **Tables** and select **New** > **New Table Definition**. The Table Definition Editor opens.
5. Type `BONUS` for the table name and click **Next**.
6. In the Table Columns page, create the following two columns:
   - memberid - BIGINT, key column
   - bonusPoint - INTEGER, nullable
7. Click **Next** twice to go to the Foreign Keys page.
8. Click **Add Another** to create a new foreign key with the following information:
   - **Foreign key name**: F_User_Bonus
   - **On Delete**: CASCADE
   - **Target table**: NULLID.USERS
9. From **Source Columns**, select **member_Id** and click the **>** button.
10. Click **Finish** to create the new table.

To create the BONUS table map, do the following:

1. In WebSphere Studio Application Developer, go to the J2EE perspective.
2. From the J2EE Hierarchy view, expand **EJB Modules**.
3. Right-click **Member-MemberManagementData EJB module** and select **Open With** > **Mapping Editor**.
4. From the upper left pane (the Enterprise Beans pane), expand **Member** and select **User bean**.
5. From the upper right corner pane, hold down the **Ctrl** key and select the **MEMBER**, **USERS** and **BONUS** tables. All three tables should be highlighted.
6. From the pull-down menu, choose **Mapping** > **Create Mapping**.
7. From the pull-down menu, choose **Mapping** > **Match by Name**.

8. From the Overview window, select **User bean**.
9. Move to the Properties window and expand **Bean to Table Strategy**.
10. Change the **Descrimator Value** from `User` to `U`.
11. Save your EJB mapping (you can press **Ctrl+S** on your keyboard).

At this point, the User bean contains errors indicating that some abstract classes are not implemented. These errors are fixed when deployed code is generated as described in "Generating deploy code."

### Generating deploy code

Since you have modified the code for the User entity bean, you must regenerate its deployed code.

To re-generate the deployed code, do the following:

1. In WebSphere Studio Application Developer, go to the the J2EE perspective.
2. From the J2EE Hierarchy view, expand **EJB Modules**.
3. Right-click **Member-MemberManagementData EJB module** and select **Generate** > **Deploy and RMIC Code**.
4. Select **All EJBs** and click **Finish**.

## Migrating a customized JSP file

In the same *WebSphere Commerce Programming Guide and Tutorials* guide tutorial, you modified the ProductDisplay.jsp template so that customers will be able to see the customized price that you added to the order process business logic. To migrate a customized JSP file to WebSphere Studio Application Developer, follow these steps:

1. Open WebSphere Studio Application Developer and switch to the Resource perspective.
2. Click **File** > **Import** > **Folder** and click **Next**.
3. Browse to the directory *VAJ_installdir*\ide\project_resources\IBM WebSphere Test Environment\hosts\default_host\default_app\web\store_directory
4. Select ProductDisplay.jsp to import.
5. Choose the folder \Stores\Web Content to contain your files.

## Testing your new logic in the Express Store sample store

Before you can verify your new business logic, you must start your server. To do this, from the Servers view, right-click **WebSphereCommerceServer** and select **Start**.

To verify your new business logic in Express Store that is running in WebSphere Studio Application Developer, do the following:

1. Once the WebSphere Commerce Server instance has started inside of WebSphere Studio Application Developer, open a browser and type the URL for your store's home page. For example, type the following URL: http://*host_name*/webapp/wcs/stores/servlet/StoreCatalogDisplay?storeId=*store_Id*&catalogId=*catalog_Id*&langId=-1

   where:

   *host_name*
   The fully qualified host name of your WebSphere Commerce Studio machine

   *store_Id*
   The identifier for your store.

   *catalog_Id*
   The identifier for your store's catalog.
2. Click the **Register** link under the **Services** heading and then click **Register** under the **New Customer** heading.
3. Register a new customer using the e-mail address `wctester@wc` and the password `wctester1`.
4. Fill in other fields with test values and click **Submit** and leave the browser open.
5. Open the DB2 Command Center and do the following:
   a. Click the **Interactive** tab.
   b. In the **Command** field, do the following:
      1) Type:
      ```
      connect to your_database_name
      ```
      where *your_database_name* is the name of your WebSphere Commerce database
      2) Click the **Execute** icon.
      3) Type:
      ```
      select users_id from userreg where logonid = 'wctester@wc'
      ```
      4) Click the **Execute** icon.
   c. The **Query Results** tab displays the entry for the customer you registered in the previous step. Make note of the customer's USERS_ID value.
   d. Update the newly registered customer's balance of bonus points.
   e. Click the **Interactive** tab and in the **Command** field, type the following:
      ```
      update BONUS set BONUSPOINT = 1000 where MEMBERID = users_id
      ```

where *users_id* is the value from step 5c on page 64.

f. Click the **Execute** icon.

6. In the browser, click the **Men's** link to view the men's fashions section of the store.
7. Click the link for the featured special to view the product page. The page displays the regular price, and the discounted price based upon the customer's balance of bonus points.

# Part 5. Appendixes

# 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.

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

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

IBM Director of Licensing
IBM Corporation
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, 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 Canada Ltd.
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
Canada

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 measurement 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 is for planning purposes only. The information herein is subject to change before the products described become available.

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.

Credit card images, trademarks and trade names provided in this product should be used only by merchants authorized by the credit card mark's owner to accept payment via that credit card.

## Trademarks

The IBM logo and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both:

DB2 IBM VisualAge WebSphere

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

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

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

Printed in USA