Zip code validation exercise

Zip code validation exercise

Contents

Zip code validation exercise 1
Introduction 1
Assumptions 1
Prerequisites 1
Creating the new enterprise bean 2
Creating and populating the database table . . 5
New command logic 6
Modifying the application configuration . . 6
Creating packages for command logic . . . 7
Adding code for error handling 7
Adding the zip code validation logic . . . 7
Registering the new command 8
Modifying a JSP template 8
Testing the zip code validation 9

Notices 11
Trademarks and service marks 14

Zip code validation exercise

Introduction

In this exercise we will look at how to create a new entity bean and extend an existing command in order to provide WebSphere Commerce with the ability to validate zip codes entered into addresses.

When entering an address into WebSphere Commerce one of the steps that happens by default is that the address information is passed to the AuditAddressCmd for validation. Out-of-the-box this command contains no logic, but with a little code we will be able to get it to verify the zip codes.

The strategy that we will be using is to create a new table to hold the data used for validation. This new table will store the zip code, city, state, and country information. The zip code and city will then be used as a key to look up the state and country. Our new command logic will use the state and country information that was retrieved to see if it matches what was entered in the address. If there is a match, the regular shopping flow continues, but if not, an error will be returned to the JSP template.

Assumptions

In the creation of this example exercise, the following simplifying assumptions were made:

1. It is assumed that a zip code is unique in a given state but may span a number of cities. To that end, a primary key constraint has been set up on the table that includes the zip code and city columns.
2. It is assumed that the zip code and city pairs are unique to a particular country.

Prerequisites

Ensure that you have satisfied the following prerequisites before starting this exercise:

- You must have installed the enhancements to IBM® WebSphere® Commerce Studio.
- You must use a DB2® database, as your development database.
- You must have published a store (based upon the FashionFlow sample store) within the development environment.
- You must locate the sample code package required for this exercise. This file is called `zipcode54.zip`. It is packaged with this document. Extract the contents of the `zipcode54.zip` file into a temporary directory. This directory will be referred to as *tempDir* in subsequent sections of this document.

Creating the new enterprise bean

Since a new database table will be created, you must create a new entity bean that corresponds to that table. A wizard within WebSphere Studio Application Developer will be used to create this new bean.

The first step in creating this new bean is to create a new EJB Project that will store the bean, as follows:

1. Open WebSphere Studio Application Developer.
2. Switch to the J2EE perspective and open up the J2EE Hierarchy view.
3. Right-click **EJB Modules** and select **New > EJB Project**.
4. In the **Project name** field enter `zipcode-ejb`
5. Select to add the new bean to an existing enterprise application. Browse to locate **WebSphereCommerce**.
6. Click **Next**.
7. Click **Finish**.

In order to create our new entity bean, we must be able to access some WebSphere Commerce classes. To do this, we add a project to the build path of the zipcode-ejb project, as follows:

1. Right-click the **zipcode-ejb** project and select **Properties**.
2. Select **Java™ Build Path**.
3. Select the **Projects** tab in the right pane.
4. Locate and select the **wcs.ccore.ejbs**.
5. Click **OK** to save the new settings.

The next step is to create a new table definition for the table that corresponds to the new entity bean, as follows:

1. Switch to the Data perspective and select the Data Definition view.
2. Locate the **zipcode-ejb** project and expand it until you can select **ejbModule > META-INF**
3. Right-click **META-INF** and select **New Database Definition**
4. In the **Database name** field, enter the name of your development database.
5. Select **Yes** to create a new directory
6. Expand the META-INF directory until you find the new database definition that was just created.
7. Right-click the database and select **New > Database schema definition**
8. In the **Schema name** field, enter `NULLID`. This forces WebSphere Studio Application Developer use the default schema name for a user. This is

important so that the enterprise bean can be installed on different databases later (for example, at deployment time).

9. Click **Finish**.
10. Expand the NULLID schema definition and select **Tables**. Right-click **Tables** and select **New > New table definition**.
11. In the **Table name** field, enter `zipcodevalidation` and click **Next**.
12. Click **Add another** to create a column, and then create the column as follows:
    a. In the **Column name** field enter `zipcode`.
    b. From the **Column type** drop-down list select `character`.
    c. Select **Key column**.
    d. Ensure that **For bit data** is *not* selected.
    e. In the **Length** field, enter `15`.
13. Click **Add another** to create a column, and then create the column as follows:
    a. In the **Column name** field enter `city`.
    b. From the **Column type** drop-down list select `character`.
    c. Select **Key column**.
    d. Ensure that **For bit data** is *not* selected.
    e. In the **Length** field, enter `30`.
14. Click **Add another** to create a column, and then create the column as follows:
    a. In the **Column name** field enter `state`.
    b. From the **Column type** drop-down list select `character`.
    c. Do not select **Key column**.
    d. Ensure that **For bit data** is *not* selected.
    e. In the **Length** field, enter `30`.
15. Click **Add another** to create a column, and then create the column as follows:
    a. In the **Column name** field enter `country`.
    b. From the **Column type** drop-down list select `character`.
    c. Do not select **Key column**.
    d. Ensure that **For bit data** is *not* selected.
    e. In the **Length** field, enter `30`.
16. Click **Finish** to create the table.

The next step is to create the new bean itself. A wizard is used to create the new enterprise bean, as follows:

1. Switch to the J2EE perspective and open up the J2EE Hierarchy view.

2. Right-click the **zipcode-ejb** project and select **New > Enterprise Bean**. The Enterprise Bean Creation wizard opens.
3. The zipcode-ejb project is already selected. Click **Next**.
4. In the Create an Enterprise Bean window, do the following:
   a. Select **Entity bean with container-managed persistence (CMP) fields**
   b. In the **Bean name** field, enter `ZipcodeValidation`.
   c. In the **Source folder** field, leave the default value that is specified (ejbModule).
   d. In the **Default package** field, enter `com.ibm.commerce.validation.objects`.
   e. Click **Next**.
5. In the Enterprise Bean Details window, do the following:
   a. Click **Add** to add a four new CMP attributes for the columns in the zip code table.
      The Create CMP Attribute window opens. In this window, do the following:
      1) In the **Name** field, enter `zipcode`.
      2) In the **Type** field, enter `String`.
      3) Select the **Key Field** check box.
      4) Click **Apply**.
      5) In the **Name** field, enter `city`.
      6) In the **Type** field, enter `String`.
      7) Select the **Key Field** check box.
      8) Click **Apply**.
      9) In the **Name** field, enter `state`.
      10) In the **Type** field, enter `String`.
      11) Click **Apply**.
      12) In the **Name** field, enter `country`.
      13) In the **Type** field, enter `String`.
      14) Click **Apply**.
      15) Click **Close** to close this window.
   b. Clear the **Use the single key attribute type for the key class** check box, then click **Next**.
6. In the EJB Java Class Details window, do the following:
   a. To select the bean's superclass, click **Browse**.
      The Type Selection window opens.
   b. In the **Select a class using: (any)** field, enter `ECEntityBean` and click **OK**. This selects the com.ibm.commerce.base.objects.ECEntityBean as the superclass.

c. Click **Finish**.

The ZipcodeValidation bean should now be visible.

The next step is to create a new access bean that corresponds to the ZipcodeValidation bean, as follows:

1. In the J2EE Hierarchy view, expand **EJB Modules**, then Right-click **zipcode-ejb** and select **New > Access Bean**.
   The Add an Access Bean window opens.
2. Select **Copy Helper** and click **Next**.
3. Select the **ZipcodeValidation** bean and click **Next**.
4. From the Constructor method drop-down list, select **findByPrimaryKey(com.ibm.commerce.validation.objects.ZipcodeValidationKey)** as the constructor method.
5. Select all attributes in the Attribute Helpers section.
6. Click **Finish**.
7. Save your work.

The next step is to create a mapping between the fields in the ZipcodeValidation enterprise bean and the database table, as follows:

1. Right-click the **zipcode-ejb** project and select **Generate > EJB to RDB Mapping**.
2. Select **Meet-in-the-Middle** and click **Next**.
3. Match by name and type.
4. Click **Finish**.
5. You should now have a mapping between the database and the table. Verify that the columns and attributes match.
6. Save your changes and close the map editor.

The final step in creating this new enterprise bean is to generate the deployed code, as follows:

1. Right-click the **zipcode-ejb** project and select **Generate > Deploy and RMIC code**.
2. Select the **ZipcodeValidation** bean and click **Finish**.

Creating and populating the database table

In order to make this new enterprise bean useful, a database table will have to be created and populated with sample data, as follows:

1. Using a DB2 command line, connect to your development database, as follows:

   ```
   db2 connect to developmentDB user dbuser using dbpassword
   ```

where *dbuser* is the database user and *dbpassword* is the user's password.

2. Issue the following command

   ```
   db2 -tf \tempDir\zipcodevalidation.ddl
   ```

   where *tempDir* is the directoy into which you extracted the zip code validation package.
3. Disconnect from the database.

The ZipcodeValidation enterprise bean should now be able to look up zip codes.

New command logic

In order to have WebSphere Commerce call our new business logic when it is adding a new address, we must extend one of the existing commands and register it in the WebSphere Commerce command registry. We will create a new command that extends the AuditAddressCmd.

The first step in creating this new command is to create a new Java project, as follows:

1. Switch to the Java perspective.
2. From the **File** menu select **New > Project**.
3. In the left pane, select **Java**. In the right pane select **Java project** and click **Next**.
4. In the **Project name** field, enter `ZipcodeValidation` and click **Next**.
5. In the Java Settings window, click the **Projects** tab and then click **Select All**.
6. Click the Libraries tab and then click **Add External JARs**.
7. Navigate to the *WSAD_installdir*\runtimes\aes_v4\lib directory (where *WSAD_installdir* is the WebSphere Studio Application Developer installation directory) and locate the j2ee.jar file. Click **Open** to add this JAR file.
8. Click **Add External JARs**again. Navigate to the *WSAD_installdir*\runtimes\aes_v4\lib directory and locate the ivjejb35.jar file. Click **Open** to add this JAR file.
9. Click **Finish**.

Modifying the application configuration

Before adding the new command logic, we will update the configuration of the existing WebSphere Commerce application to include both the new EJB project and the new Java project that will contain the new command logic.

To update this configuration information, do the following:

1. Switch to the J2EE perspective and select the J2EE navigator.

2. Expand the **WebSphereCommerceServer** folder, then expand the META-INF folder.
3. Double-click the **application.xml** file. The Application Deployment Descriptor opens.
4. Select the Module tab.
5. Click **Add** under the list of modules. Select the **zipcode-ejb.jar** and then click **Finish**.
6. Click Add under the list of Project Utility JARs.
7. Select the **ZipcodeValidation** project and then in the **URI** field enter `lib/ZipcodeValidation.jar`. Click **Finish**.
8. Select the **wcstores** project. Expand **wcstores > Web Content >META-INF > MANIFEST.MF**.
9. Right-click the **MANIFEST.MF** and select **Open with > Text editor**.
10. In the new editor add `lib/ZipcodeValidation.jar` to the end of the file.
11. Repeat steps 8 through 10 for the wctools project.

Creating packages for command logic

The next step is to create packages that will hold the command logic and associated files, as follows:

1. Switch to the J2EE perspective and select the J2EE navigator.
2. Right-click the **ZipcodeValidation** project and select **New > Package**.
3. In the **Package name** field, enter `com.ibm.commerce.validation.constants` as the package name and click **Finish**.
4. Right-click the **ZipcodeValidation** project and select **New > Package**.
5. In the **Package name** field, enter `com.ibm.commerce.validation.commands` as the package name and click **Finish**.

Adding code for error handling

Now we will add the code that will be used by the error handler, as follows:

1. Right-click the **com.ibm.commerce.validation.constants** package and select **New > Class**.
2. In the **Class name** field, enter `ValidationConstants` as the class name and click **Finish**.
3. Into the source code for the new class, add the following code:

```
final static public String INVALID_ZIPCODE = "INVALID_ZIPCODE";
```

4. Press Ctrl+S to save and then close the editor.

Adding the zip code validation logic

The next step is to add the new logic that performs the validation of the zip code, as follows:

1. Right-click the **com.ibm.commerce.validation.commands** package and select **New > Class**.

2. In the **Class name** field, enter `AuditZipcodeCmdImpl` as the class name and click **Finish**.
3. Next locate the AuditZipcodeCmdImpl.java file in the zip code validation package and cut and paste the contents of the class into your command. Overwrite all code that was generated with the code you paste in.
4. Press Ctrl+S to save and then close the editor.
5. Next you need to manually build the Java project. Right-click the **ZipcodeValidation** project and select **Build Project**.

What this class does is to pull the zip code, city, state, and country fields from the address. These fields are stored in the requestProperties object and are populated by other code. The command then uses the zip code and city to create an entity bean to retrieve the state and country info. After doing a compare, if the state and country pair do not match then the command will throw a new ECApplicationException and return the new error code that was created in the ValidationConstants class. The view JSP template is responsible for displaying an error to the user.

Registering the new command

The last thing to do before the command can be used is to register it with the database. Included in the zip code validation package is a database script that will register the new command.

To register the new command, do the following:

1. Using a DB2 command line, connect to your development database, as follows:

   ```
   db2 connect to developmentDB user dbuser using dbpassword
   ```

   where *dbuser* is the database user and *dbpassword* is the user's password.
2. Issue the following command

   ```
   db2 -tf \tempDir\LoadWCRegistry.sql
   ```

   where *tempDir* is the directory into which you extracted the zip code validation package.
3. Disconnect from the database.

Modifying a JSP template

One JSP template needs to be modified so that it will display an error if the user enters an invalid zip code.

To modify the JSP template, do the following:

1. Switch to the Web perspective and locate the wcstores project.

2. Expand **wcstores > Web Content > *FashionFlow_name*** where *FashionFlow_name* is the name of your store based upon the FashionFlow sample store.
3. Next expand ***FashionFlow_name*** **> UserArea > AccountSection > AddressbookSubsection**.
4. Double-click the **AddressForm.jsp** file.
5. You must add a new import statement to this file. Locate the other import statements near the top of the file immediately after the existing statements, add the following:

   ```
   <%@ page import="com.ibm.commerce.validation.constants.*" %>
   ```

6. Search in this file and locate the line that contains `ERROR_MESSAGE8`. As the next line, add the following:

   ```
   if (strErrorCode.equals(ValidationConstants.INVALID_ZIPCODE))
     strErrorMessage = "Zipcode is not valid for city/state/country entered";
   ```

7. Save your changes.

Testing the zip code validation

The next step is to use the local test environment to test your new enterprise bean and command logic, as follows:

1. Switch to the Server perspective.
2. Right-click **WebSphereCommerceServer** and select **Start**.
3. After the server has started, open up a browser and open up the FashionFlow store. For example, in a Web browser, enter the following URL:

   ```
   http://localhost/webapp/wcs/stores/servlet/FashionFlow/index.jsp
   ```

4. Register a user or log in as a registered user.
5. Go to the address book and create a new address. For the first test case, enter an invalid address. You should receive an error.
6. Now enter a zip code that is in the database table that was created. For example, one valid set of values is Beverly Hills (city), 90210 (zip code), California (state), USA (country) 8. This should allow you to add the address.

Notices

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

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

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
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 information 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.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

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.

COPYRIGHT LICENSE:

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

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

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

Trademarks and service marks

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
WebSphere

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc.

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

Printed in U.S.A.