Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support

Skill Level: Intermediate

Maria G. Querales (<u>querales@us.ibm.com</u>) Software Developer, IBM®

Kevin D. Kelley (<u>kdkelley@us.ibm.com</u>) Information Developer, IBM

Shahin Mohammadi-Rashedi (shahin@us.ibm.com) IMS™ SOA Demonstration Team Lead

IBM Corporation 15 March 2009

Abstract

In this tutorial you will use IBM Rational® Developer for System z® Software Version 7.5 to build a J2C JavaTM Bean based on the input and output formats defined in the MFS source of an IMS transaction. Once the Java Bean is created, you will reuse the invoke and use the IMS business logic from a Web application such as JavaServer Pages component (JSP).

About this tutorial

This tutorial will take you through the steps of using the Java EE Connector (J2C) component of IBM Rational Developer for System z and IBM IMS MFS SOA Support and will help you to familiarize yourself with the Java Platform Enterprise Edition (Java EE, previously known as J2EE) development environment and the J2C. You will be working with the sample Phonebook IMS transaction (IVTNO), which is one of the IMS installation verification programs shipped with IMS.

Rational Developer for System z

Is a superset of Rational Application Developer. The Java EE perspective, J2C wizard and Web tools that you will use in this tutorial are also available in Rational Application Developer.

Rational Application Developer

Simplifies application development for enterprise information systems (EIS) such as IMS by providing wizard-based tools and a list of adapters ready to use. Rational Application Developer provides J2C wizards that enable you to create J2C applications, either as standalone programs or as added function to existing applications.

Java EE Connector Architecture

Java EE Connector Architecture (JCA) is a standard for connecting a Java-based technology solution for connecting application servers and EIS.

J2C JavaBeans™

A JavaBean that communicates with an EIS through JCA.

J2C Application

A typical J2C application consists of a J2C JavaBean with one or more methods that call EIS functions. For IMS, the input and outputs to these functions are data binding classes. After you have created a J2C JavaBean, you can create Web pages, an EJB, or a Web service for the J2C JavaBean.

MFS

Message Format Service (MFS) enables application programmers to specify screen formats, application input and output fields, and various device characteristics that define the end-user interface to an IMS transaction. While MFS manages device-specific information, the IMS transaction defines the application logic.

IMS MFS SOA Support

Next generation of MFS Web services support introduced in IBM Rational Application Developer for WebSphere® Software Version 7.5 (Rational Application Developer). Designed to use the latest programming model, J2EE Connector (J2C) that is supported in Rational Application Developer.

Objectives

To gain hands-on experience extending IMS MFS-based applications to the Web as a part of a Web page. Software tools that are part of Rational Developer for System z and IMS MFS SOA Support make the transformation processes easy, as the tutorial will demonstrate.

Upon completion of this tutorial, you will be able to:

- Use Rational Developer for System z and its built-in J2C tools.
- Enable an IMS MFS-based application as J2C JavaBean
- Generate a JSP for a J2C JavaBean

System requirements for the tutorial:

For purposes of this demo, you will be using IBM Rational Developer for System z Software Version 7.5 with IBM WebSphere Application Server Version 7. If you decide to test MFS SOA Support in your local environment, use the following software and platform:

- Software installed on Microsoft® Windows®
 - Rational Developer for System z Software Version 7.5.0 iFix001
 - WebSphere Application Server Version 6.1
- System software installed on IBM z/OS®
 - IMS Version 9 or Version 10
 - IMS Connect Version 9 or Version 10
 - o OTMA
 - TCP/IP

Checklist for the first time implementation

You may find it helpful to have the following checklist available before proceeding with your own implementation for the first time.

	Your environment	This tutorial:
MFS Source file(s)	This can be obtained from	C:\mfs_source_file\dfsivf1.mfs
	IMS application	
	programmers.	
IMS Connect host name (or IP address)	This can be obtained from	Host name:
and port number.	IMS system programmers.	ZSERVEROS.DFW.IBM.COM
		Port number: 9999
IMS Data store.	This can be obtained from	IMSC
	IMS system programmers.	
Workspace directory and project name	A naming standard is	C:\Workspaces7.5\SANDBOX
will be used by Rational Developer for	recommended.	
System z when generating artifacts.		

Overview of development tasks

To complete this tutorial you will perform the following tasks:

Task 1: Create the J2C JavaBean for the IMS MFS-based transaction

For the first task of the lab you will create a J2C JavaBean that is based on the input and output definitions of an MFS-based IMS application. The definitions reside in the MFS source file of the IMS application (you will import this MFS source file as part of task 1). You will use these definitions to populate the service definition, including the XML schema, for both input and output types.

Task 2: Create a JSP for the J2C JavaBean

For task 2 you will use a JSP file to embed the J2C JavaBean that you created in Task 1. You need to embed the J2C JavaBean into a JSP because a J2C JavaBean is like a method call... it is not an application. The JSP is the actual application that displays things as a Web page.

What is a JSP? A Java Server Page (JSP) is a Java technology that allows software developers to dynamically generate HTML, XML or other types of documents in response to a Web client request.

Task 3: Test the J2C application using the JSP on IBM WebSphere Application Server

In Task 3 you test what you have built. You will use the JSP you created in Task 2 (TestClient.jsp) to test the J2C application on IBM WebSphere Application Server.

Figure 1 shows how to create and test a J2C application for an IMS MFS-based transaction using IMS MFS Support.



Figure 1. Using RDz and system z to accomplish lab objectives

Task 1: Create the J2C JavaBean for the IMS MFS-based transaction



KEY POINTS

What are you about to do in Task 1? For the first task of the lab you will create a J2C JavaBean that is based on the input and output definitions of an MFS-based IMS application. These definitions reside in the MFS source file of the IMS application and they will be used to populate the service definition, including the XML schema for both input and output types.

Item	Description
MFS Source file	The MFS Source file (dfsivf1.mfs) defines the structure of the input and output messages of the IMS IVP Phonebook application.
MID	MFS input formatting occurs when a message input descriptor (MID) name is provided with an input message.
MOD	The MFS message output descriptor (MOD) can supply a MID name to be used for formatting the next input message.

1.1 M The IBM Rational Developer for System z V 7.5 is started and you are using the Workspaces7.5 \SANDBOX.



Important!

This Lab uses the directory C:\Workspaces7.5\SANDBOX as your work workspace.



What is a workspace?

A workspace is a directory that stores files for your projects. You can select your own directory or take the default directory. Artifacts created by Rational Developer for System z will be stored in this directory.



1.2 ► If the Welcome page displays you can close it from the Welcome tab. Open the Java EE perspective on the Rational Developer for System z workspace.



Figure 3. Setting the Java EE perspective

1.3 ► You can also close the Welcome to z/OS projects panel, if it is displayed. The z/OS Projects – IBM Rational Developer for System z workspace displays.



Figure 4. The IBM Rational Developer for System z workspace

What is Java EE? The Java Platform, Enterprise Edition (previously known as Java 2 Platform, Enterprise Edition, or J2EE) provides a standard for developing component-based, multi-tier, enterprise applications. A Java EE application system typically includes the following tiers: Client tier: In the client tier, Web components, such as servlets and JSPs, or standalone Java applications provide a dynamic interface to the middle tier. Middle tier: In the server tier, or middle tier, enterprise beans and Web Services encapsulate reusable, distributable business logic for the application. These server-tier components are contained on a Java EE Application Server, which provides the platform for these components to perform actions and store data. Enterprise data tier: In the data tier, the enterprise's data is stored and persisted, typically in a relational database. J2EE applications are comprised of components, containers, and services. Web components, such as servlets and JSPs, provide dynamic responses to requests from a Web page. EJB components contain server-side business logic for enterprise applications. Web and EJB component containers host services that support Web and EJB modules.

1.4 From the Rational Developer for System z workspace, click File > New > Other to start the wizard.

Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support © Copyright IBM Corporation 1994, 2009. All rights reserved.

File Edit Navigate Sea	rch Project Run	Cl
New	Alt+Shift+N	Enterprise Application Project
🟠 Accept Team Invitation		🔯 Dynamic Web Project
Open File		ST EJB Project
Close	Ctrl+W	Connector Project
Close All	Ctrl+Shift+W	Application Client Project
Save .	Ctrl+S	Project
Save As Save All Revert	Ctrl+Shift+S	🖹 Work Item 🎦 Folder
Move		File
Rename	F2	
Refresh	F5	📑 Other 🗼 Ctrl+N
Convert Line Delimiters	То	.
🖶 Print	Ctrl+P	
Switch Workspace Restart		•
≧∎ Import ≧ Export		
Properties	Alt+Enter	
Exit		

Figure 5. The menu path for selecting a wizard



In this section you will create a Java bean that communicates with IMS using J2C. Here we will define whether the Java bean is a managed or non-managed resource, along with defining the TCP/IP address, port and IMS datastore name. The Select a wizard page opens.

© New	
Select a wizard Create a J2C bean that communicates with an Enterprise Information System through the J2EE Connector Architecture.	
<u>W</u> izards:	
type filter text	
 Enterprise Service Tools Example EMF Model Creation Wizards GMF-Xpand Graphical Modeling Framework MS J2C CICS/IMS Java Data Binding Command Bean U2C Bean Web Page, Web Service, or EJB from J2C Java Bean 	
Show All Wizards. (?) < Back	Cancel

Figure 6. The Select a wizard page

1.5 Expand the J2C folder and select the J2C Bean wizard. Click Next. The Resource Adapter Selection page opens.



1.6 Expand the IMS TM folder and choose the IMS Connector for Java (IBM: 9.1.0.2.5a) for your project. Click Next. The Connector Import page opens.

New J2C Bean		×
Resource Adapter Selection Choose a resource adapter that will commu System). Choose the resource adapter: Choose the resource adapter:	nunicate to the EIS (Enterprise Information	
ECIResourceAdapter (IBM : 5.1.0 ECIResourceAdapter (IBM : 6.0.2 ECIResourceAdapter (IBM : 7.1.0 ECIXAResourceAdapter (IBM : 9.1.0 ECIXARESOURCE (IBM : 9.1.0	.0.3) .2.2) .0.2) 7.1.0.2) 9.1.0.1.5b) 9.1.0.2.5a) 1: 10.2.0) Expand the IMS TM folder and select the IMS Connector for Java (IBM: 9.1.0.2.5.1), and click Next	
IMS Connector for Java	<u>More details about J2C bea</u>	<u>in</u>
⑦ Sac	ck Next > Finish Cancel	

Figure 7. The Resource Adapter Selection page



What is a resource adapter?

Resource adapters allow your application to communicate with the enterprise information system (EIS). A resource adapter is a system-level software driver that is used by a Java application to connect to an EIS. The resource adapters reside on the application server and provide connectivity between the EIS, the application server, and the enterprise application. Applications deployed on the application server communicate with the resource adapter using the Common Client Interface (CCI). The RAR file (a type of archive file format) contains all the information necessary for installing, configuring and running a Resource Adapter. Resource Adapters comply

Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support © Copyright IBM Corporation 1994, 2009. All rights reserved. with the J2EE Java EE Connector Architecture (**J2C**). In this lab we are using IMS TM Resource Adapter to connect to IMS.



IMS TM Resource Adapter Version 10 is based on the newer JCA 1.5 standards, and therefore it is located under the 1.5 section.

Rational Developer for System z supports Java EE Connector Architecture (JCA) version 1.0 and version 1.5.

New J2C Bean	
Connector Import Import a connector resource adapt	ter archive from the file system.
Connector file: C:\Program F Connector project:* imsico91025 Target server: WebSphere A	Files\IBM\SDP\ResourceAdapters\ims15\imsico91025a.rar a Application Server v7.0 Select WebSphere Application Server v7.0 and click Next
0	< Back Next > Finish Cancel

Figure 8. The Connector Import page

1.7 ► Select WebSphere Application Server v7.0 runtime environment from Target server, and click Next. The Scenario Selection page opens.

New J2C Bean	
Scenario Selection Select the type of IMS applications that the new J2C Java bean will access.	0-
O IMS COBOL, PL/I or C-based applications Generate a J2C Java bean that accesses COBOL, PL/I and C based IMS applica	tions
• IMS MFS-based applications Browse and import MFS metadata from MFS sources to generate a J2C Java b your MFS-based IMS application	ean for
Select IMS MFS-based applications and click Next	
? < Back	Cancel

Figure 9. The Scenario Selection page

1.8 M Select IMS MFS-based applications and click Next. The Discovery Configuration page opens.

New J2C Bean		
Discovery Configuration The wizard can now guide you	through discovery of objects to communicate with.	5
Connection Configuration	ery.	
MFS files:*		Add Remove
Device characteristic table: MFS properties Host code page: *	Cp037 EBCDIC United States	Browse
MFS file format:	text	
Text code page: *	Cp1252 Windows Latin 1 ng desired	Select
0	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish	Cancel

Figure 10. The Discovery Configuration page

1.9 Click Add and navigate to the folder where the dfsivf1.mfs source file is (C:\mfs_source_file).

Open			? 🔀
Look in: My Recent Documents Desktop My Documents	dfsivf1.mfs	Navigate to the folder where you select the dfsivf1.mfs source file, and click Open	
My Network	File name: Files of type:	*.mfs	Open Cancel

Figure 11. Finding the MFS Source file



The MFS Source file dfsivf1.mfs defines the structure of the input and output messages of the IMS IVP Phonebook application.

1.10 Select the dfsivf1.mfs source file and click Open to return to the Discovery Configuration page.

New J2C Bean		
Discovery Configuration		
The wizard can now guide you Specify settings to begin discov	through discovery of objects to communicate with. ery.	<u>ó</u>
Connection Configuration		
C:\mfs source file\dfsivf1.	mfs	Add
	Select the MFS source file	Remove
	and click Next.	
Device characteristic table:		Browse
MFS properties		
Host code page: *	Cp037 EBCDIC United States	Select
MFS file format:	text 💌	
Text code page: *	Cp1252 Windows Latin 1	Select
Specify the level of the logging	ng desired	
0	< Back Next > Finish	Cancel

Figure 12. Click Next on the Discovery Configuration page



The dfsivf1.mfs source file is parsed through the MFS Importer and returns an operation that contains both MID and MOD names. The operation displays on the Object Discovery and Selection page.

1.11 M Accept all the defaults and click Next. The Object Discovery and Selection page opens.

© New J2C Bean			
Object Discovery and Selection			
Expand nodes to view and select discovered objects. Use find number of objects displayed.	iltering, if necessary, to limit		
Objects discovered by query:	Objects to be imported:		
₽ Z □			
Select RunIVTNOMI1 in the Objects discovered by query panel and click the ">" to import it into the project.			
	> Finish Cancel		

Figure 13. The Object Discovery and Selection page

1.12 Select RunIVTNOMI1 in the Objects discovered by query panel and click the ">" to import it into the project. The Configuration Parameters for RunIVTNOMI1 page opens.

💽 New J2C Bean		X
Configuration Para Set the configuration pa	meters for 'RunIVTNOMI1' arameters, then press OK.	0
Business object name:	RunIVTNOMI1	-
MID:	IVTNOMI1	
Logical page number:	1	
MOD:*		\sim
IVTNO Device type:	3270,1	Add Remove
Device feature:	Ignore 💌	
InteractionSpec proper	ies	
Interaction verb:	SYNC_SEND_RECEIVE (1)	
IMS request type:	IMS_REQUEST_TYPE_MFS_TRANSACTIC	
Execution timeout:	5	
Commit Mode:	SEND_THEN_COMMIT (1)	
Advanced >>		•
?	ОК	Cancel

Figure 14. The Configuration Parameters for RunIVTNOMI1

1.13 Click Add. The Add Value page opens.

• Add Value	
Select values from DFSM01 DFSM02 DFSM03 DFSM05	the list below: Select all of the MOD names and click OK.
•	OK Cancel

Figure 15. The Add Value page

Select the following MOD names and click OK:

- DFSMO1
- DFSMO2
- DFSMO3
- DFSMO5

Tip: You can select all of the MOD names at one time.



What are MID and MOD?

MFS input formatting occurs when a message input descriptor (MID) name is provided with an input message. The MFS message output descriptor (MOD) can supply a MID name to be used for formatting the next input message.

C onfiguration Para Set the configuration p	meters for 'RunIVTNOMI1' arameters, then press OK.	0
Business object name:	RunIVTNOMI1	_
MID:	IVTNOMI1	
Logical page number:	1	-
MOD:* IVTNO DF5M01 DF5M02 DF5M02	The MOD you selected display. Click OK.	Add
Device type:	3270,1	-
Device feature:	Ignore	-
InteractionSpec proper	ties	
Interaction verb:	SYNC_SEND_RECEIVE (1)	3
IMS request type:	IMS_REQUEST_TYPE_MFS_TRANSACTIC	2
Execution timeout:	0	
Commit Mode:	SEND_THEN_COMMIT (1)	-
Advanced >>		
0	ОК	Cancel

Figure 16. The MODs you selected are included in the Configuration Parameters page

1.14 Accept the defaults on the Configuration Parameters for 'RunIVTNOMI1' page and then click OK to return to the Object Discovery and Selection page.

New J2C Bean	
Object Discovery and Selection Expand nodes to view and select discovered objects. In number of objects displayed.	Use filtering, if necessary, to limit
Objects discovered by query:	Objects to be imported:
₽ <u>□</u>	
RunIVTNOMI1	RunIVTNOMI1
⑦ Sack	Next > Finish Cancel

Figure 17. The Object Discovery and Selection page shows RunIVTNOMI1 added into the Objects to be imported panel.

1.15 M Click Next. The J2C Bean Creation and Deployment Configuration page opens.

Figure 18. The J2C Bean Creation and Deployment Configuration

1.16 From the J2C Bean Creation and Deployment Configuration page, click New next to the Project name field to create a new project into which the J2C JavaBean will be generated. The New

Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support © Copyright IBM Corporation 1994, 2009. All rights reserved. 22 of 44 Source Project page displays.

• New Source Project		
New Source Project Creation Select project type.		
 Java project Web project EJB project 	Select Java project and click Next.	
⑦ Sack	Next > Finish	Cancel

Figure 19. The New Source Project Creation page



Why Java project?

we will use it as the container for the J2C JavaBean source. Java projects are not defined in the Java SE specification. They are used as the lowest unit to organize the workspace and contain all resources needed for a Java application as images, source, class and properties files.

The J2C Wizard gives a choice between creating Java, Web or EJB project. We choose Java project because

1.17 M Select Java project and click Next. The Create a Java Project page opens.

New Java Project		
Create a Java Project Create a Java project in the workspace or	in an external location.	
Project name: MFSPhonebook Contents		Enter MFSPhonebook in the Project name field, and click Finish.
 Create new project in workspace Create project from existing source 		
Directory: C:\IMS Project\MFSPhoneb	ook	Browse
JRE		
Use default JRE (Currently 'jdk')	I ma	Configure JREs
O Use a project specific JRE: O Use an execution environment JRE:	Jdk JavaSE-1.6	*
Project layout		
O Use project folder as root for source	s and class files	
• Create separate folders for sources	and class files	Configure default
Working sets		
Warking sets:		Select
0	<pre> < Back Next ></pre>	Finish Cancel

Figure 20. The Create a Java Project page

1.18 Enter MFSphonebook in the Project Name field, and click Finish to return to the J2C Bean Creation and Deployment Configuration page.mt

New J2C Bean			
J2C Bean Creation and Deployment O JNDI Lookup Name: cannot be empt	Configuration γ.		
Save properties			
Project name: *	MFSPhonebook		Browse New
Package name: 🛛 💉	com.ibm.ims.mfs.soa	>	Browse New
Interface name: 🧹	PhoneBook		
Implementation name: *	PhoneBookImpl	1	
Generate Command Bean			
Generate Command Bean for met Enable generate Command IVTNOMI1:	hod: IVTNOMI1 Bean	Enter the fields as Note that the implementation fie for you	shown. e Id will fill
Command Bean name:	IVTNOMI1CommandBean		
Command Bean input name:	input		
Command Bean output name:	output		
Connection properties Managed connections are obtained Non-managed connections are obt Managed Connection (recomme	d through JNDI name lookup tained directly from the resc ended)	from the application se purce adapter.	rver.
JNDI Lookup Name: *			Browse New
Non-managed Connection			
	Click here to s Look	tart the JNDI Name up wizard.	
Click to launch J2C deployment wiza	rd		
0	< Back	Next > Finis	Cancel

Figure 21. The J2C Bean Creation and Deployment page

1.19 Enter com.ibm.ims.mfs.soa in the Package name field and PhoneBook in the Interface name field. The Implementation name will auto-complete for you. Click New under Connection properties. The JNDI Lookup Wizard displays

JNDI Lookup Wizard	
Server selection Select the server on which to deploy the resource adapter.	
Server: WebSphere Application Server v7.0 at localhost	New
⑦ < <u>Back</u> <u>Next</u> > <u>Finish</u>	Cancel

Figure 22. The Server selection page

11

Ī

What is an Application Server?

An application server is a component-based product that resides in the middle-tier of a server centric architecture. It provides middleware services for security and state maintenance, along with data access and persistence.

What is JNDI?

The **Java Naming and Directory Interface (JNDI)** is an API for directory service that allows clients to discover and look up data and objects via a name. In this lab we assign a unique JNDI name to our managed connection. Our J2C JavaBean will then use this JNDI name to look up the connection on the WebSphere Application Server.

1.20 Miclick Next and the New J2C Connection Factory page opens.

JNDI Lookup V	Wizard 📃 🗖 🔀		
New J2C Connection Factory Create a new J2C Connection Factory for the resource adapter selected at the first page of the J2C wizard.			
Resource Adapter: JNDI Name: Connection class name:	IMS Connector for Java (IBM : 9.1.0.2.5a)		
Host name: ZSERVEROS.DFW.IBM.COM Port Number: 9999 Local option IMS Connect name:			
Data store name: *	IMSC Fill out the fields as shown. Ensure that TCP/IP is checked, and click Finish.		
0	< <u>Back</u> <u>Next</u> > <u>Einish</u> Cancel		

Figure 23. The JNDI Lookup Wizard



Our application will interact with the IMS TM resource adapter through an object called a **connection factory**. IMS connection factories are used to create pre-configured connections to the **IMS transaction manager** (IMS TM).

When an application uses the IMS TM Resource Adapter, it interacts with IMS using connections between the IMS TM resource adapter and IMS Connect that are created by the IMS TM Resource Adapter. These connections can be **managed** or **non-managed**.

1.21 Enter MyJNDIname in the JNDI Name field. Ensure that **TCP/IP** is checked (default) and enter the required connection information as indicated by the asterisk (*):

Host name:	ZSERVEROS.DFW.IBM.COM
Port:	9999
Data store name:	IMSC

1.22 M Click Finish to return to the J2C Bean Creation and Deployment Configuration page.

Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support © Copyright IBM Corporation 1994, 2009. All rights reserved. 27 of 44

• New J2C Bean		
J2C Bean Creation and Deployment Specify the properties for creating and	t Configuration d running the J2C bean.	
Save properties		
Project name: *	MFSPhonebook	Browse New
Package name: *	com.ibm.ims.mfs.soa	Browse New
Interface name: *	PhoneBook	
Implementation name: *	PhoneBookImpl	
Generate Command Bean	¥	
Enable generate Command IVTNOMI1:	Bean	
Command Bean name:	IVTNOMI1CommandBean	
Command Bean input name:	input	
Command Bean output name:	output	
Connection properties Managed connections are obtaine Non-managed connections are ob	d through JNDI name lookup from the application se tained directly from the resource adapter. ended)	rver.
JNDI Lookup Name: 🌂	MyJNDIname	Browse New
Non-managed Connection	urd	
0	< Back Next > Finis	h Cancel

Figure 24. The J2C Bean Creation and Deployment Configuration page

1.23 N Click Finish.

Managed and Non-Managed connections

A **managed connection** runs inside a Web application server. With a managed connection, the application server provides transaction management and connection pool management, and the application server can send security information. In addition, managed connections allow connection information to be maintained by the system administrator. As connection information changes (the type of communications, the port, etc.), the system administrator can adjust the connection characteristics and no Java objects need to be regenerated.



A **non-managed connection** is designed to run where connections management supplied by an application server is not available. The characteristics of the connection must be specified and are hard-coded into the generated object. You can change the connection characteristics from your program, but you will need to generate your J2C code appropriately. Because non-managed connections are not always convenient to change, and they do not take advantage of the connection pooling, transaction management, and security management that are provided by an application server, it is easy to see why managed connections are recommended.

Java EE - MFSPhonebook/src/com/ibm/ims/mfs/soa/PhoneBookImpl.java - IBM F File Edit Source Refactor Navigate Search Project Run ClearCase Window Help Open Project 1 • 🗐 💩 🗄 😭 • 🚳 🗄 💷 • 🗆 🖳 🕅 Close Project 約 和 やや や Build All CtrI+B 😤 Ente 🖾 🧏 😫 Servi 🛱 Tea - -J Pho Build Project 日雪碧 p Build Working Set 🕀 🔛 imsico91025a Cl an... () in MFSPhonebook **Build Automatically** 1 RemoteSystemsTempFiles Apply Patch ... E) Generate Javadoc... DIname" Convert to a Dynamic Web project ... hector2.ims.i Properties public class PhoneBookImpl implements com.ibm.ims.mfs. private ConnectionSpec typeLevelConnectionSpec; private InteractionSpec invokedInteractionSpec; private InteractionSpec interactionSpec; private ConnectionSpec connectionSpec; private Connection connection; private ConnectionFactory connectionFactory; 1** * @j2c.interactionSpec class="com.ibm.connector2. * @j2c.interactionSpec-property name="imsRequestT * @j2c.interactionSpec-property name="mapName" va * @generated *1

For this lab example, we will use a managed connection to IMS,

Figure 25. Select Project > Clean...

1.24 \bowtie Click Project > Clean The Clean pane comes up

Create a J2C application for an IMS MFS-based transaction using IMS MFS SOA Support © Copyright IBM Corporation 1994, 2009. All rights reserved.



Figure 26. The Clean Pane

1.25 M Click OK.

You just completed Task 1: "Create the J2C JavaBean for the IMS MFS-based transaction"
 What is next? Now that you have created the J2C JavaBean, you can proceed to generate other J2EE resources, such as JSP based on this J2C Javabean.
 Number of tasks completed: 1
 Number of tasks left: 2



KEY POINT

What are you about to do in Task 2? For task 2 you will generate a JSP file in which you will embed the J2C JavaBean you created in Task 1. You need to embed the J2C JavaBean into a JSP because a J2C JavaBean is like a method call... it is not an application. The JSP is the actual application that displays things as a Web page.

In this task you will create a Simple JSP file (Web pages) to embed the previously generated J2C JavaBean.

2.1 → From the IBM Rational Developer for System z workspace, select File > New > Other. The Select a wizard page opens.



Figure 27. The Select a wizard page with the Web page, Web service, or EJB from J2C Java Bean item selected 2.2 Expand the J2C folder and select the Web page, Web service, or EJB from J2C Java Bean item, and click Next. The Java bean selection pane opens.

O Java EE Resourc	e from J2C Java Bean	
J2C Java bean selection Choose the J2C Java Bean create a Java EE Resource	implementation class that you want to for.	
J2C bean implementation:	\MFSPhonebook\src\com\ibm\ims\mfs\soa\PhoneBookImpl.java This is the implementation of the PhoneBook application. Click Next.	Browse
0	< Back Next > Finish	Cancel

Figure 28. The implantation of the PhoneBook application is selected

2.3 Click Next. The Deployment Information page opens.



Figure 29. The Deployment Information page



JSP technology enables you to generate dynamic web content, such as HTML, DHTML, XHTML, and XML files, to include in a Web application. JSP files are one way that the product implements server-side dynamic page content. JSP files allow a Web server, such as WebSphere Application Server, to add content dynamically to your HTML pages before they are sent to a requesting browser. **EJB (Enterprise JavaBeans)** and **Web Services** are other powerful architectures that can interface with our J2C JavaBean. Web Services are covered in the optional Task 4 of this lab.

2.4 M Select Simple JSP and click Next. The Simple JSP Creation page opens.

Java EE Resource from J2	C Java Bean	
Simple JSP Creation Choose to create simple JSPs with default	input fields.	
Web project:*		Browse
0	< Back Next > Finish	Cancel

Figure 30. The Simple JSP Creation page

2.5 Click New. The Dynamic Web Project page opens.

New Dynamic Web Project			
Dynamic Web Project Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.			
Project name: MFSPhoneBookWeb Project contents: Use default Directory: C:\IMS Project\MFSPhoneBookWeb Target Runtime WebSphere Application Server v7.0		Browse	
Dynamic Web Module version 2.5 Configuration	Pull down and select WebSphere Application Server v7.0	·	
Default Configuration for WebSphere Application Server v7.0 ✓ Modify A good starting point for working with WebSphere Application Server v7.0 runtime. Additional facets can later be installed to add new functionality to the project. EAR Membership ✓ Add project to an EAR ✓ New EAR Project Name: MFSPhoneBookWebEAR ✓ New			
O Sack	Next > Finish	Cancel	

Figure 31. The Dynamic Web Project page

2.6 ► Enter MFSPhoneBookWeb in the Project Name field, and from the pull-down menu, select WebSphere Application Server v7.0 for the Target Runtime. Verify that Add project to EAR box is checked and click Finish. The Simple JSP Creation page opens.

O Java EE I	Resource from J2C Java Bean	
Simple JSP Crea Choose to crea	a tion te simple JSPs with default input fields.	
Web project:* JSP folder: Advanced >>	MFSPhoneBookWeb MyDSP Enter MyJSP for the JSP folder name and click Finish	Browse New
0	< Back Next > Finish	Cancel

Figure 32. The Simple JSP Creation page

2.7 M Name the JSP folder MyJSP and click Finish. Notice that the MFSPhoneBookWeb web project and the PBSimpleJSPEAR items are created in the Project Explorer panel.



Figure 33. MFSPhoneBookWeb web project and the PBSimpleJSPEAR items are created in the Project Explorer panel

You have completed Task 2: "Create a JSP for the J2C JavaBean" What is next? Test the J2C application using the JSP Number of tasks completed: 2 Number of tasks Left: 1

Task 3: <u>Test the J2C Application using the JSP</u> on IBM WebSphere Application Server



What are you about to do in Task 3? All that you have to do now is test. In Task 3 you will start WebSphere Application Server, add your project to the application server runtime environment, and test your application using the simple JSP client that you created.

3.1 M In the Project Explorer panel on the left, expand MFSPhoneBookWeb > WebContent > MyJSP.



Figure 34. Expand MyJSP and right click TestClient.jsp

3.2 \bowtie Right click TestClient.jsp. Select Run As > Run on Server.

New		Þ	
Open	F3		
Open With	10000	×	
📄 Сору	Ctrl+C		
Paste	Ctrl+V		
🔀 Delete	Delete		
Move			
Rename	F2		
🔄 Import			
Export			
🖹 Refresh	F5		
Links		ĸ	
Validate			
Software Analy	yzer	×.	
Compile JSPs			
Run As		Þ	🚦 1 Run on Server 👘 Alt+Shift+X, I
Debug As			Run Configurations
Profile As		۰Ļ	run comga danon
Team		×.	
Compare With		×.	
Replace With			
Page Template	e	×.	
Source		×.	
Properties	Alt+Ente	r	



The Run On Server page opens.

Run On Server		
Run On Server		
Select which server to use		
How do you want to select the server?		
Choose an existing server		
O Manually define a new server		
Select the server that you want to use:		
type filter text		
Iocalhost WebSphere Application Server v7.0 at localho Started		
WebSphere Application Server v7.0		
? < Back Next > Finish	Cancel	

Figure 36. Run on Server Page

3.3 Accept the defaults and click Finish. It will take a moment for the server to start. The workspace will display separate panels for Methods, Inputs, and Result. This may take a few minuets. ?

🚺 PhoneBookImpl.java 🛛 🕥	Technology Quickstarts 🛛 🥹 Web Services Test Client 🖾
🗘 📫 😹 http://localhos	t:9081/MFSPhoneBookWeb/MyJSP/TestClient.jsp
Methods <u>IVTNOMI1</u> (com.ibm.ims.mfs.e	Select a method to test.
	Result result: N/A

Figure 37. The workspace displays Methods, Inputs, and Result in separate panels in the workspace

3.4 ▶ Under Methods, click IVTNOMI1.

Methods	Inputs
• <u>IVTNOMI1</u> (com.ibm.ims.mfs.en	arg: eXTUnicode0023: cMD: nAME2: nAME1: zIP: recordShortDescription: recordName: Invoke Clear

Figure 38. The PhoneBook interface opens in the Inputs panel.

- 3.5 ▶ Enter Display in the Cmd field (command field). 3.6 ▶ Enter LAST1 in the Name1 field (last name field).

3.7 ► Click Invoke. The results display at the bottom of the Inputs panel. Scroll down in the Results panel to see that the entry is displayed.

Methods	Inputs	
• IVTNOMI1	arg:	
(com.ibm.ims.mfs.en	eXTUnicode0023:	
	cMD:	Display
	nAME2:	
	nAME1:	SMITH
	zIP:	
	recordShortDescription	
	recordName:	
	Invoke	
		15,-7]
	sEGNO: recordName:	0019
	mSG:	ENTRY WAS DISPLAYED
	eXTUnicode0023:	5639
	nAME2:	JOHN
	nAME1:	SMITH
	cMD:	DISPLAY
< >	recordShortDescri	ption. com.ibm.ims.mfs.cmd.databinding.IVTNO
	Figure 39. The results	display in the Result panel

3.8 W You can click Clear to empty the fields so you can enter another name.



Congratulations! You have completed the Lab!

Resources Learn

- If you want to learn more about MFS SOA Support there is an article in the IMS Newsletter: <u>http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.imsnews.doc/</u> <u>newsletters/v803/v803.htm#article4</u>
- More information about MFS SOA can be found at the IMS Integration Suite page: <u>http://www-01.ibm.com/software/data/ims/toolkit/</u>
- More information about MFS SOA can be found in the IMS Information Center: <u>http://</u> publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.etools.mfs.doc/mfs.htm

Get products and technologies

- Download trial versions of IBM Rational software.
- Download a trial version of <u>IBM Rational Developer for system z</u>.

Discuss

- <u>Participate in the discussion forum</u>.
- Check out <u>developerWorks® blogs</u> and get involved in the <u>developerWorks community</u>.

About the authors

Maria G Querales is a Software Developer at IBM Kevin D. Kelley is an Information Developer at IBM Shahin Mohammadi-Rashedi is an IMS SOA Demonstration Team Lead at IBM

Trademark notice

IBM, the IBM logo and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.