



IBM Integration Bus

Pattern Authoring Lab 6

Groups and Table Parameters

June, 2013

Hands-on lab built at product code level Version 9.0

1.	INTRODUCTION
2.	CREATING THE PATTERN4
3.	TESTING AND DEBUGGING THE PATTERN

1. Introduction

In this lab you will use 3 new Pattern Parameters features introduced in WebSphere Message Broker version 8.

This lab has been tested using IBM Integration Bus V9.0:

- 1. Parameter Group Hiding
- 2. Project Name Parameter
- 3. Table Parameter

The objective of the lab is to create a Routing Pattern that will allow you to generate a dynamic routing message flow. The pattern will take the number of routing destinations (MQOutput nodes) and the routing table as parameters. For each destination, the Pattern Configuration Editor will show one parameter group with its corresponding parameters.

You will also define a parameter to be used as the Project Name for the generated Pattern instance.

2. Creating the Pattern

1. Create a new Application by clicking on "New Application". (The example below shows an empty workspace, but may be different if you already existing items in the workspace).

Enter "PatternParameters" as the application name (this exact name is required, since it will match some java code later in the lab). Click Finish.

	😥 New Application	_ 0
Application Development	Create a new application	
New Application	An application is a deployable container that provides isolation at runtime. Enter a name for the new application.	A
New Integration Service		
<u>New</u> Library	Application name PatternParameters	
	? Finish	Cancel
	👹 Start from samples	

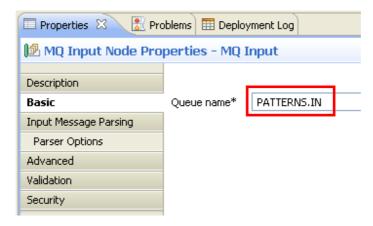
- 🔚 Application Dev 🛛 🗙 Retterns Explor 🖉 🕞 $\overline{\nabla}$ 雪 Application Development New.. - A PatternParameters New Artifact Create a new artifact. Message Flow 비접 <u>Subflow</u> Message Model ... 🚖 Message Map ESQL File Decision Service Broker Schema Adapter Connection Database Definition 💐 Data Lineage Documents BAR file 🔣 Integration Bus Test Client
- 2. Create a new Message Flow by clicking on New -> Message Flow under the PatternParameters application.

3. Enter "RoutingFlow" as the message flow name. Click Finish.

🜔 New Message Flo	w						
Create a new message flow							
Select a container fo	r the new message flow						
Container:	PatternParameters		New				
Message flow name:	RoutingFlow						
Flow organization -							
Use default bro	ker schema						
Schema; (default	broker schema)		7				
			1				
0		Finish	Cancel				

- 🖽 RoutingFlow.msgflow 🔀 👌 🔅 Palette 📐 🖌 🖉 🙀 Favorites 🖳 WebSphere MQ 🔚 JMS 💭 HTTP 🧟 Web Services 👰 SCA . 🐻 WebSphere Adapters 🕞 Routing 🛅 Transformation Construction Route MQ Input 🗋 Database
- 4. Drop an MQInput node and a Route node onto the canvas. Connect them as shown:

5. Set the MQInput node's queue name to "PATTERNS.IN".



6. Click on the Route node, and select the properties tab.

On the Basic tab, click on the Add button to insert a new Filter Pattern.

Enter "true()" as the Filter Pattern and click OK. Leave the Routing output terminal to "Match".

		1			Add Filter table Filter pattern* true() Routing output termina Match		Edk		
Graph User De	X 🔝 Probl	lems 🎞 C				_		1	° (
Description	🔕 Filter tab		ite must be set for this prope	rty.		ОК	Cancel		
Basic	Filter table*		Filter pattern	Routir	ng output terminal			Add	1
Monitoring	-							Edit Delete	
								2 7	

Note that the only purpose of this step is to avoid the error that would appear if the Filter table was left empty.

7. Press Ctrl+S or File->Save to save the message flow.

Check that the Problems view has no errors (apart from the "no outbound connections" warning).

8. Create a new Pattern Authoring project by clicking File->New->Pattern Authoring Project.

Enter "RoutingPattern" as the Pattern name and "RoutingPatternProject" as the Project name.

Click Next.

🖸 New Pattern Authoring Project
Create a New Pattern Authoring Project This wizard creates a new Pattern Authoring project.
Pattern name: RoutingPattern
Project name: RoutingPatternProject
Use default location
Location: C:\student\workspace\RoutingPatternProject Browse
(<u>Back Next</u> > <u>Finish</u> Cancel

9. Select the "PatternParameters" project, and click Finish.

🜔 New Pattern Authoring Pro	oject						
Create a New Pattern Authoring Project Specify dependencies on other projects.							
Referenced projects:							
PatternParameters							
?	< <u>B</u> ack	Next >	Einish	Cancel			

10. When the Pattern Authoring Editor opens, click on the "Pattern Configuration" tab.

Click on the "Add Group" button.

B RoutingFlow.msgflow	Pattern Specification	🗄 *RoutingPattern.pattern 😣
🖪 Pattern Config	juration	
		iate the pattern parameters with their target properties. end your pattern with <u>Java and PHP code</u> .
Groups and Parameter	s	
		Add Group
		Edit
		Delete
		Enumerated Types
		Tables
		Expand All
		Collapse All
Java and PHP code Using pattern parameters properties in message flow	and your own logic, you can ext vs. Use PHP to generate text file	tend your pattern with Java and PHP code. Use Java to access and update the nodes, connections, and es, for example ESQL and MQSC scripts.
		Add
		Edit
		Delete
Source Files Pattern Configu	uration Categories Create Patt	tern

11. Enter "General Parameters" as both the Group Display Name and Description.

Click OK.

	🖬 Add Group	X
Configure an XPath expression that controls when this group is enabled in the Pattern Instance editor. Basic Enable Group Display Display name: General Parameters Description: General Parameters Group Options Select this option to create help information for this group in the pattern documentation.	Configure group	
Basic Enable Group Display Display name: Description: General Parameters Group Options Generate help documentation Select this option to create help information for this group in the pattern documentation.	Configure the pattern parameter g	oup and how it is displayed to pattern users.
Group Display Display name: General Parameters Description: General Parameters Group Options Group Options Select this option to create help information for this group in the pattern documentation.	Configure an XPath expression that	controls when this group is enabled in the Pattern Instance editor.
Group Display Display name: General Parameters Description: General Parameters Group Options Group Options Select this option to create help information for this group in the pattern documentation.	Basic Epable	
Display name: General Parameters Description: General Parameters Group Options Select this option to create help information for this group in the pattern documentation.		
Group Options Generate help documentation Select this option to create help information for this group in the pattern documentation.		General Parameters
Group Options Generate help documentation Select this option to create help information for this group in the pattern documentation.	Description	Concerted Darameters
Generate help documentation Select this option to create help information for this group in the pattern documentation.	Description.	General Paradilecers
	Group Options	
Display parameters in a group box Select this option to display this group, and any parameters it contains, with a surrounding box.	Generate help documentation	Select this option to create help information for this group in the pattern documentation.
	✔ Display parameters in a group	box Select this option to display this group, and any parameters it contains, with a surrounding box.
OK Cancel		OK Cancel

12. Now add three parameters to the "General Parameters" group: Project Name, OutputsAmount and RoutingTable.

Click on the "General Parameters" group and click on the "Add Parameter" button.

🕮 RoutingFlow.msgflow 🛛 📲 Pattern Specification 🕴 🔡 *RoutingPattern.pattern 😫	
R Pattern Configuration	
Configure your groups and pattern parameters and associate the pattern parameters with their target properties. Using pattern parameters and your own logic, you can extend your pattern with <u>Java and PHP code</u> .	
Groups and Parameters	
General Parameters	Add Group
	😫 Add Parameter
	Edit
	Delete
	Enumerated Types
	Tables
	2 😺
	Expand All
	Collapse All

13. Enter "Project Name" as the Display Name, and "projName" as the Parameter ID. Click OK.

🚔 Add Parameter		×
Configure pattern parameter		
Configure the pattern parameter and how i	t is displayed to pattern users.	
Basic Editor Transform Enable		
Parameter Display Display name: Project Name	Parameter ID: projName	
-Parameter Options		
Hide the parameter	Select this option to hide the parameter and to use an XPath expression to set the value of the parameter when a pattern instance is created.	
Mandatory parameter	Select this option if the pattern user must enter a value for the parameter. Mandatory parameters also display a field prompt to guide the pattern user.	
Field prompt: Enter your para	ameter value	
parameter HTML file.		
>Describe the parameter here	Describe the parameter here	
	ОК Салсе!	

14. Click on the "Add Parameter" button again, and enter "OutputsAmount" as the Display Name, and "outAmount" as the Parameter ID.

nfigure pattern p	oarameter		
onfigure the pattern p	parameter and how it i	is displayed to pattern users.	
sic Editor Transf	orm Enable		
Parameter Display — Display name:	OutputsAmount	Parameter ID: outAmount	
biopidy ridinor	o depacor iniciality		
Parameter Options —			
Hide the paramete		Select this option to hide the parameter and to use an XPath expression to set the value	of the parameter
Hide the parameter	ər	when a pattern instance is created.	
Mandatory param	leter	Select this option if the pattern user must enter a value for the parameter. Mandatory pa a field prompt to quide the pattern user.	arameters also display
Field prompt:	Enter your param	neter value	
Do not include any <	html> or <head> tag:</head>	display as help text for this parameter. Preview parameter help s because the text is inserted into a	
Enter any HTML or te	html> or <head> tag:</head>	Isplay as help text for this parameter. Preview parameter help s because the text is inserted into a	
Enter any HTML or te Do not include any <	html> or <head> tag:</head>	display as help text for this parameter. Is because the text is inserted into a	
Enter any HTML or te Do not include any ⊲ parameter HTML file.	html> or <head> tag:</head>	s because the text is inserted into a	
Enter any HTML or te Do not include any ⊲ parameter HTML file.	html> or <head> tag:</head>	s because the text is inserted into a	OK Cancel

15. Then click on the Editor tab.

Select "Drop Down Selection" as the Parameter Editor.

Click on the Enumerated Types button to define the possible values of the parameter.

🚔 Add Parameter		X
Configure pattern parameter		
📀 No enumerated types are available for t	his pattern parameter.	
Basic Editor Transform Enable		
Parameter Editor		
Parameter editor:	Drop Down Selection	Configure Editor
Type selection:	Enumerated Types	Tables
Default value:		~
Dependencies		
Pattern parameters can depend on one of the message types that are available in the	r more parameters. For example, a message type parameter depends on a message set parameter depends on a message set p	arameter because it displays
Dependencies are configured automatical		
	orrectly, the dependent parameter (for example, the message type parameter) must be in	the same group and be
after the parameter on which it depends.	an easy, and appendent parameter (for example) and message type parameter, mass be m	rate same group and be
This parameter depends on the following	parameters:	
		OK Cancel

16. In the "Configure Enumerated Types" window, click on the Add button to create a new enumerated type.

Enter "outputsAmount" as the Enumeration name.

Cli	ck	OK.	

Configure enumerated types Configure the enumerated types that are used by the pattern parameters. An enumerated type is a set of values that a pattern parameter accepts. The display names are shown in a list for pattern users when they create an instance of this pattern. The value is the content that is stored in the target property and can be alphanumeric or '_' only.		
The display names are shown in a list for pattern users when they create an instance of this pattern.		
Enumerated type: Add Remove Rename	Duplicate	
Display Name		
D Enter New Enumeration	Add	
Enter a name for the enumeration:	Remove	
OK. Cancel		
	Reset Values	5
You cannot remove an enumerated type if it is being used by a parameter or is defined by a target property. Change the type of a pattern parameter by clicking Edit in the Pattern Parameters tab. This enumerated type is used by the following parameters:		

17. Enter the following values in the table using the Add button for each row.

To edit the values, click on the row, delete the existing value (e.g. "Display Name1"), overtype with the new value, and press Enter to make the change.

Click OK.

Configure Enumerated	Гурез			×	
Configure enumerated types					
Configure the enumerated types that are used by the pattern parameters.					
The display names are shown in a	alues that a pattern parameter accepts a list for pattern users when they creat tored in the target property and can be	e an instance of this	pattern. only.		
Enumerated type:	outputsAmount			▼	
			Add Remove	Rename Duplicate	
Display Name		Value			
1 2		1 2		Add	
3		3		Remove	
5		5			
				Reset Values	
You cannot remove an enumerated type if it is being used by a parameter or is defined by a target property. Change the type of a pattern parameter by clicking Edit in the Pattern Parameters tab. This enumerated type is used by the following parameters:					
				OK Cancel	

18. Back in the "Add Parameter" window, click OK.

19. Now you will add the Routing Table parameter. To do this you will use a new type of parameter, which allows you to enter tables as parameters of the pattern.

Click on the Add Parameter button again, and enter "Routing Table" as the Display name, and "RoutingTable" (no spaces) as the Parameter ID.

🚔 Edit Parameter: Ro	uting Table				×
Configure pattern pa	arameter				
Configure the pattern pa	rameter and how it is o	lisplayed to pattern users.			
Basic Editor Transfor	m Enable				
Parameter Display					
Display name:	Routing Table	Paramet	er ID: Rou	utingTable	
Parameter Options —					
		Select this option to hide the param	eter and to use an XPath e	expression to set the value	e of the parameter
Hide the parameter	•	when a pattern instance is created.			
Mandatory parame	ter	Select this option if the pattern use a field prompt to guide the pattern		e parameter. Mandatory p	parameters also display
Field prompt:	Enter your paramet	er value			
Help Text (HTML)					
Do not include any <hi< td=""><td>t that you want to disp tml> or <head> tags b</head></td><td>lay as help text for this parameter. ecause the text is inserted into a</td><td>Preview parameter help</td><td></td><td></td></hi<>	t that you want to disp tml> or <head> tags b</head>	lay as help text for this parameter. ecause the text is inserted into a	Preview parameter help		
parameter HTML file.					
Describe the par	ameter here	~	Describe the para	ameter here	<u>~</u>
			Describe are pure		
		*			~
					OK Cancel

20. Then click on the Editor tab to select the appropriate Editor for this parameter.

Select "Table Editor" as the Parameter editor.

Then click on the Tables button to create the required table.

≌ Add Parameter			
Configure pattern parameter			
😣 No tables are available for this patte	rn parameter.		
Basic Editor Transform Enable			
Parameter Editor			
Parameter editor:	Table Editor	~	Configure Editor
Type selection:	×	Enumerated Types	T <u>a</u> bles
Default value:			~
Dependencies			
Pattern parameters can depend on on the message types that are available i	e or more parameters. For example, a message type parame n the selected message set.	eter depends on a message set p	arameter because it displays
Dependencies are configured automati	ically by the Pattern Authoring editor.		
after the parameter on which it depen This parameter depends on the followi			
			OK Cancel

21. Click on the Add button and enter "RoutingTable" as the table name.

Click OK.

ole		×
table:		
	OK Cancel	
	ble table:	table:

22. Add two columns (two rows in the table) using the Add button, to reflect the Route node Filter table, like shown, with the following values:

Column Name	<u>Column ID</u>
FilterPattern	filterPattern
Output Terminal	terminal

onfigure tables						
Configure the tables	that are used by the pattern para	ameters.				
table is a collection of ach row in the table co	rows which a pattern user config ontains one or more columns of da	ures. ta,				
Table:	RoutingTable	1	• <u>A</u> dd	<u>R</u> emove	Rename	Dyplicate
Column Name		Column Id	Width			
FilterPattern Output Terminal		filterPattern terminal				A <u>d</u> d
output reminal			_			Remove
	xt that you want to display as hel html> or <head> tags because th</head>					
		e text is inserted into a parameter HTI	ML file.			A
hange the type of a pa	able if it is being used by a parame attern parameter by clicking Edit i	r.	YL file.			4
hange the type of a pa		r.	ML file.			×
hange the type of a pa	attern parameter by clicking Edit i	r.	ML file.			×
hange the type of a pa	attern parameter by clicking Edit i	r.	ML file.			×

Then click OK.

23. In the Add Parameter window, click OK.

24. Then you are going to add 5 groups, one per each possible MQOutput node the user can generate with the pattern.

Click on the Add Group button, enter "MQOutput1" as the Display name and Description.

🖪 Add Group		×
Configure group Configure the pattern parameter group and how it i Configure an XPath expression that controls when I	is displayed to pattern users. this group is enabled in the Pattern Instance editor.	
Basic Enable Group Display Display name: MQOutput1 Description: MQOutput1		
Group Options		
Generate help documentation	Select this option to create help information for this group in the pattern documentation.	
Display parameters in a group box	Select this option to display this group, and any parameters it contains, with a surrounding box.	
	ок	Cancel

Click OK.

25. Select the MQOutput1 group and click on the Add Parameter button.

Enter "Queue Name" as the Display name and "qn1" as the Parameter ID.

Click Ok.

Pattern Authoring - Groups and Table Parameters Provided by IBM BetaWorks

- 26. Repeat the last step to create a parameter with "Queue Manager" as the Display name and "qm1" as the Parameter ID.
- 27. Save your progress pressing Ctrl+S or File->Save.
- 28. Create a new group by clicking on the Add Group button.

Enter "MQOutput2" as the Display Name and Description.

dd Group	
nfigure group	
onfigure the pattern parameter group and he onfigure an XPath expression that controls w	ow it is displayed to pattern users. when this group is enabled in the Pattern Instance editor.
sic Enable	
Group Display	
Display name: MQOutp	out2
Description: MQOutp	ut2
Group Options	
Generate help documentation	Select this option to create help information for this group in the pattern documentation.
Display parameters in a group box	Select this option to display this group, and any parameters it contains, with a surrounding box.

This time, click on the "Enable" tab.

29. Now you are going to define when should this group be shown.

The objective is to show the MQOutput2 group only if the OutputsAmount was set to a value greater than 1.

Double-click on the "OutputsAmount" parameter (under General Parameters group)

This will insert the "pp:getValue("outAmount") sentence in the Expression field.

Append ">1" to the expression.

🖥 Add Group			
Configure group Configure the pattern parameter group and how it is displayed to p Configure an XPath expression that controls when this group is en		ce editor.	
Basic Enable If the XPath expression evaluates to true, the group is enabled an The values of pattern parameters within a disabled group do not p	nd shown, otherwise it is dis populate target properties in	abled and hidden. n an instance of the user-defined pattern.	
Functions		Operators	
e ≑t: Boolean e ÷t: Number e ÷t: Pattern e ÷t: String		-+ -* -dv -= -!= -<	
Function name:	₩ Use	Operator:	Use
Groups and Parameters	Parameter ID	Test Value	
General Parameters Project Name	projName		
📔 OutputsAmount	outAmount	1	
E Routing Table	RoutingTable		
Test value: 1	🔒 Set 🛛 Par	ameter ID: outAmount	Use
Expression Evaluation Expression: pp:getValue('outAmount')>1 Result:			▶ Evaluate
		ОК	Cancel

Note that "outAmount" was used instead of "OutputsAmount" because the first is the Parameter ID while the last one is the Parameter display name.

30. You can test your expression by using the Evaluate button.

Enter "1" as the Test Value and click Set. Then click the Evaluate button.

📧 Add Group			×
Configure group Configure the pattern parameter group and how it is displayed to Configure an XPath expression that controls when this group is er	pattern users. abled in the Pattern Instanc	te editor.	
Basic Enable If the XPath expression evaluates to true, the group is enabled an The values of pattern parameters within a disabled group do not p	nd shown, otherwise it is dis populate target properties in	abled and hidden. an instance of the user-defined pa	attern.
Functions		Operators	
 ⊕ *t Boolean ⊕ *t Number ⊕ *t Pattern ⊕ *t String 		+ 	
Function name:	🐉 Use	Operator:	Use
Groups and Parameters	Parameter ID	Test Value	
□ Image: General Parameters ^{\fmupped} Project Name ^{\fmupped} OutputsAmount ^{\fmupped} Routing Table Image: MQOutput1	projName outAmount RoutingTable	1	
Test value: Expression Evaluation	Set Para	ameter ID: outAmount	
Expression: pp:getValue('outAmount')>1 Result: Disabled (false)			Evaluate
			OK Cancel

You can see the result of the evaluation in the Result field.

In this case, the Result is Disabled, because OutputsAmount value was set to 1.

31. Now set the Test Value to "2", click the Set button. Click on the Evaluate button.

🖬 Add Group			×
Configure group			
Configure the pattern parameter group and how it is dis Configure an XPath expression that controls when this g	olayed to pattern users. roup is enabled in the Pattern Insta	nce editor.	
Basic Enable			
If the XPath expression evaluates to true, the group is e The values of pattern parameters within a disabled grou	enabled and shown, otherwise it is o p do not populate target properties	isabled and hidden. in an instance of the user-defined patter	n.
Functions		Operators	
B →t Boolean B →t Number B →t Pattern B →t String		-+ 	
Function name:	Use	Operator:	Use
Groups and Parameters	Parameter ID	Test Value	
🖃 💽 General Parameters			
Project Name	projName		
	outAmount	2	
	RoutingTable		
Test value: 2	Set Pa	rameter ID: outAmount	Use
Expression Evaluation			
Expression: pp:getValue('outAmount')>	1		Evaluate
Result: Enabled (true)			
			OK Cancel

Check that the Result now is Enabled, because OutputsAmount value was greater than 1.

Click OK.

32. Add two parameters to the MQOutput2 group (Queue Name and Queue Manager) the same way you did before, with these values:

Display Name	Parameter ID
Queue Name	qn2
Queue Manager	qm2

33. Repeat 3 times the steps 29-30 with the following values:

Group Display name	Enable Expression
MQOutput3	pp:getValue('outAmount')>2
MQOutput4	pp:getValue('outAmount')>3
MQOutput5	pp:getValue('outAmount')>4

Then add two parameters to each group with the following values:

Group	Display Name	Parameter ID
MQOutput3	Queue Name	qn3
MQOulputs	Queue Manager	qm3
MQOutput4	Queue Name	qn4
MQOulpul4	Queue Manager	qm4
MQOutput5	Queue Name	qn5
MQOulputs	Queue Manager	qm5

It should look like this:

Groups and P	arameters
	eral Parameters
	Project Name (projName)
	OutputsAmount (outAmount)
	Routing Table (RoutingTable)
📄 🗐 MQ(Output1
	Queue Name (qn1)
	Queue Manager (qm1)
🚊 🖳 🔂 MQ	Dutput2
	Queue Name (qn2)
	Queue Manager (qm2)
📄 📴 MQ4	Dutput3
	Queue Name (qn3)
	Queue Manager (qm3)
📄 📴 MQ0	Output4
	Queue Name (qn4)
	Queue Manager (qm4)
📥 🖳 🚾 MQ(Output5
	Queue Name (qn5)
	Queue Manager (qm5)

34. Press Ctrl+S or File->Save to save your Pattern.

35. Now you are going to create a Java class that will create the MQOutput nodes dynamically (according to the OutputsAmount parameter) and will associate the table parameter with the Router node Filter table.

In the "Java and PHP code" section (at the bottom of the Pattern Configuration tab, click on the Add button.

Java and PHP code Using pattern parameters and your own logic, you can extend your pattern with Java and PHP code. Use Java to access and updat properties in message flows. Use PHP to generate text files, for example ESQL and MQSC scripts.	e the nodes, connections, and
	<u>A</u> dd
	Ed <u>i</u> t
	Delete
	2 3

This will start the "Add Code" wizard.

36. Leave the "Type of code" field as "Java", and click on the "New Project" button.

-	Add Code		×
	Add code to your pattern Select the code that is called wh	en a pattern instance is generated.	
		rchive, your Java and PHP code projects are automatically packaged with your pattern a and PHP code, click "New Project".	plug-ins.
	Type of code:	Java	~
	Project name:	×	New Project
	Plug-in ID:		
	Ensure that the Java class is exp	displays the Java classes that you can use. orted from the code project. can be used in your pattern, click "New Java Class".	Θ
	Class name:		New Java Class
	You can choose to write the outp	nen a pattern instance is created. out from the PHP file into a file in a pattern instance project. ed projects into a PHP file in your code project, click "Create From File".	
	PHP file name:	×	Create From File
	Write the output from the PH		
	Output project name:	PatternParameters	¥
	Output file name:		
			OK Cancel

37. Change the java Class name to MyRoutingJava.

Uncheck the "Add PHP support to the project" option since you won't need it.

Click Finish.

I	🖸 New Pattern Authoring Java and PHP Project
1	New pattern authoring Java and PHP project 🖉
	Configure the pattern authoring Java and PHP project.
	This wizard creates a pattern authoring project.
1	The project is a plug-in that you package and distribute with your pattern plug-ins. The Plug-in ID uniquely identifies the plug-in when it is loaded by the WebSphere Message Broker Toolkit.
I	The Plug-in ID is also used to name the project in your workspace.
	Dive is ID:
	Plug-in ID: com.your.company.domain.RoutingPatternProject.code
1	Java
- -	Pattern authoring code projects always support Java. Use Java to manipulate message flows in pattern instance projects. Java can also implement other custom logic required by your pattern.
:	Add an example pattern authoring Java class to the project
-	Package name: com.vour.company.domain.RoutingPatternProject.code
-	Class name: MyRoutingJava
e	
è	PHP
	Use PHP to generate text files, such as MQSC and ESQL scripts, in pattern instances projects. If you enable this option, a templates directory and PHP examples are added to your project.
F	Add PHP support to the project
	Cancel

38. Back in the "Add Code" window, leave the default value for the Java Class name (it should have been completed automatically with the Java class generated by the wizard)

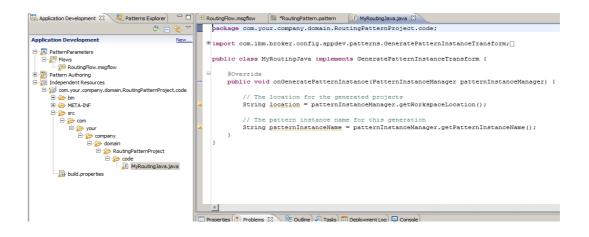
Click OK.

Add Code			
Add code to your pa	ittern		
Select the code that is c	alled when a pattern instance is generated.		
Java or PHP			
	attern archive, your Java and PHP code projects are automatically packaged with you our Java and PHP code, click "New Project".	r pattern pl	ug-ins.
Type of code:	Java		•
Project name:	com.your.company.domain.RoutingPatternProject.code	•	New Project
Plug-in ID;	com.your.company.domain.RoutingPatternProject.code		
Ensure that the Java clas	e" field, displays the Java classes that you can use. is is exported from the code project. ss that can be used in your pattern, click "New Java Class".	1	Θ
Class name:	com.your.company.domain.RoutingPatternProject.code.MyRoutingJava	•	New Java Class
You can choose to write	runs when a pattern instance is created. the output from the PHP file into a file in a pattern instance project.		æ
	eferenced projects into a PHP file in your code project, click "Create From File".		
PHP file name:	eterenced projects into a PHP file in your code project, click "Create From File".	<u> </u>	Create From File
	eterenced projects into a PHP file in your code project, click "Create From File".	<u> </u>	Create From File
		¥	Create From File
Write the output from	n the PHP file into an output file:	¥ [Create From File

39. In the Application Development view, under Independent Resources, expand the newly created Java Project (com.your.company.domain.RoutingPatternProject.code).

Navigate to com.your.company.domain.RoutingPatternProject.code -> src ->com -> your -> company -> domain -> RoutingPatternProject -> code

Double-click the MyRoutingJava.java file to open it in the Java Editor.



40. Replace the automatically generated Java Class skeleton with a pre-built version.

Right-click on the MyRoutingJava.java file and select Import.

🗟 Application Development 🛛 🖧 Patterns	Explorer 🗖 🗖	RoutingFlow.
	🖑 🖻 😫 🏹	package
Application Development	<u>New</u>	• import
PatternParameters		public
🕀 🐮 Pattern Authoring	New	•
⊡…? Independent Resources ⊡…? com.your.company.domain.RoutingPa ⊡…? bin	Open Open With	•
	Сору	
⊡ 🔂 src	Paste	
🖻 🖓 🔂 com	Delete	
⊡… 🧽 your ⊡… 🎓 company	Move Rename	
⊡ 🦢 company ⊡ 🔂 domain ⊡ 🔂 RoutingPatter	Add Bookmark	
code	import ™Export	
build.properties	Rebuild Project	
	Refresh	
	Validate Launch Universal	Test Client

41. Select General -> File System and click Next.

42. Click on the Browse button and navigate to "C:\student\PatternAuth\resources".

Select the MyRoutingJava.java file.

Check "Overwrite existing resources without warning" and click Finish.

🜔 Import				
File system Import resources from the local file system	n.			
From directory: C:\student\PatternAuth	n\resources	Create_mqsc CreateLogTa CreateLT.bal CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl CreateLT.ddl MyRoutingJa PatternAuthi	ble.bat t l.php php.txt .java.txt a2.txt wa.java Basic_Exemplar.	
Filter Types Select All De: Into folder: com.your.company.domain	select All] 🗐 PatternsAdv.] 🏹 PatternsAdv.] 🧐 PatternsAuth	ancedStart.zip ancedStartV8.zip n_Start.zip	
Overwrite existing resources without	: warning			
Create complete folder structure Advanced >>				
?	< Back	Next >	Finish	Cancel

Pattern Authoring - Groups and Table Parameters Provided by IBM BetaWorks

43. Take a few moments to review the java code and read the comments.

In the first part of the code, the message flow, the Route node, the Routing Table, the OutputsAmount parameter, the Router node Filter table are loaded into memory.

In the last line of the screenshot, the row of the Route node Filter table (the one required by the compiler) is removed.

```
public void onGeneratePatternInstance(PatternInstanceManager
patternInstanceManager) {
      //Get the RoutingFlow.msgflow message flow
      MessageFlow mf1 =
      patternInstanceManager.getMessageFlow("PatternParameters",
      "RoutingFlow.msgflow");
      //Get the Route Node object
      RouteNode routeNode = (RouteNode) mfl.getNodeByName("Route");
      //Get the Parameter Table defined in the Pattern
      PatternParameterTable paramtable =
patternInstanceManager.getParameterTable("RoutingTable");
      Integer outNo;
      //Get the "outAmount" parameter (amount of desired outputs
      outNo =
Integer.valueOf(patternInstanceManager.getParameterValue("outAmount"));
      PatternParameterRow paramTableRow;
      //Get the Filter table from the route nodes
      RouteNode.FilterTable filterTable = (RouteNode.FilterTable)
routeNode.getFilterTable();
       //delete default row in the routeNode filter table
      filterTable.removeRow(filterTable.getRows().get(0));
```

44. Then in the second portion of the code, there's a cycle that associates each row of the parameter table (RoutingTable) to the Router node Filter Table.

This portion also contains the code that creates the MQOutput nodes (with its properties taken from the pattern parameters) and connects them to the Router node.

```
for(int i=1;i<=outNo;i++) {</pre>
//Add a row to the Filter Table with the Filter Pattern and Terminal for
            011 011
paramTableRow = paramtable.getRow(i-1);
RouteNode.FilterTableRow newRow = filterTable.createRow();
newRow.setFilterPattern(paramTableRow.getValue("filterPattern"));
newRow.setRoutingOutputTerminal(paramTableRow.getValue("terminal"));
filterTable.addRow(newRow);
//Create an MQOutput
MQOutputNode mqOutNode = new MQOutputNode();
//Set MQOutput node name
mqOutNode.setNodeName("MQOutput " + i);
//Set MQOutput node Queue Name
mqOutNode.setQueueName(patternInstanceManager.getParameterValue("qn"+i));
//Set MQOutput node Queue Manager
mqOutNode.setQueueManagerName(patternInstanceManager.getParameterValue("q
m"+i));
//Set location of new node on flow editor
mqOutNode.setLocation(400,50*i);
mf1.addNode(mqOutNode);
//Add terminal to the Route Node
OutputTerminal term =
routeNode.getOutputTerminal(paramTableRow.getValue("terminal"));
//Connect Route node to MQOutput node
mf1.connect(term, mqOutNode.INPUT_TERMINAL_IN);
             }
      }
```

45. Insert a Breakpoint in the Java code by double-clicking on the blue column, in the first line of the class:



46. Close the MyRoutingJava.java file and return to the RoutingPattern.pattern in the Pattern editor.

Click on the "Source Files" tab, and select the "PatternParameters" project.

Click on the Edit button.

Page 37

Pattern Authoring - Groups and Table Parameters Provided by IBM BetaWorks 47. In the "Edit Project" window, click on the Select button next to the "Pattern parameter name" field.

Project and how it is configured when a pattern instance is generated. A Pattern expression to control when this project is generated in a pattern instance. C Create Project Options Pattern users will modify this project If this option is selected, files in this project will not be overwritten when a pattern instance is re-generated. Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance Select a pattern parameter for the project name when the pattern instance is generated.	and a second	eters_broker			
a an XPath expression to control when this project is generated in a pattern instance. C Create Project Options Pattern users will modify this project Pattern users will modify this project Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance Select a pattern parameter for the project name when the pattern instance is generated.	figure project				
c Create Project Options If this option is selected, files in this project will not be overwritten when a pattern instance is re-generated. P Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance name when it is generated. Select a pattern parameter for the project name when the pattern instance is generated: Select a pattern parameter for the project name when the pattern instance is generated:	onfigure the project and how it is	s configured when a pattern instar	nce is generated.		
Project Options Pattern users will modify this project If this option is selected, files in this project will not be overwritten when a pattern instance is re-generated. Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance name when it is generated. Select a pattern parameter for the project name when the pattern instance is generated:	e an Arachexpression to contro	n when this project is generated in	ra pattern instance.		
Project Options Pattern users will modify this project If this option is selected, files in this project will not be overwritten when a pattern instance is re-generated. Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance name when it is generated. Select a pattern parameter for the project name when the pattern instance is generated:	sic Create				
Pattern users will modify this project If this option is selected, files in this project will not be overwritten when a pattern instance is re-generated. Prefix the project name with the pattern instance name Select this option and the project name will be prefixed with the pattern instance name when it is generated. Select a pattern parameter for the project name when the pattern instance is generated: Select a pattern parameter for the project name when the pattern instance is generated:					
Instance is re-generated. Instance is re-generated. Instance is project name will be prefixed with the pattern instance name when it is generated. Select a pattern parameter for the project name when the pattern instance is generated:					
name when it is generated.	Pattern users will modify this	project		es in this project will not be overwritten v	when a pattern
	Prefix the project name with	the pattern instance name	Select this option and the pr name when it is generated.	roject name will be prefixed with the pat	tern instance
	Select a pattern parameter for t	he project name when the pattern	n instance is generated:		
Project Name Clear Select					
	Pattern parameter name:	Project Name		Clear	Select
OK Care					Cancel

Click OK.

48. In the "Select Pattern Parameter" window , select the "Project Name" parameter under the "General Parameters" group and click OK, and OK again.

🗜 Select Pattern Parameter	×
Select Pattern Parameter	
Select the pattern parameter to use for the project name when the pattern instance is generated	
	OK Cancel

49. Select the Categories tab and click on the "Add Category" button.

Enter "Routing" as the category name and click OK.

🔚 Application Development 🛛 🤻 Patterns Explorer 🛛 🗖	🗄 RoutingFlow.msgflow 🔡 *RoutingPattern.pattern 🗙	
⁽¹⁾ ⁽²⁾ ⁽²⁾ ⁽²⁾	je Categories	
Application Development <u>New</u>		
AlternParameters AlternParameters Big Rows RoutingFlow.msgflow Different Authoring Independent Resources	Create new categories and assign your pattern to a category. Double-click your category of	r pattern to edit its HTML specification.
는 🥵 com.your.company.domain.RoutingPatternProject.code 한 🗁 bin	Categories	
B B META-INF C Com C Com D C Com D C Company D C Company D C Company D C Company D C Company	Application Integration	A pattern is a reusable solu
e ← coungraterinityett e ← code MyRoutingJava.java		<pre>x ed approach to solvin ecture, design, or dep ular context. commonly recurring so that you want to achi that is being addres</pre>
		cancel that is being addres , and any constraints common usage and the applic
	Source Files Pattern Configuration Categories Create Pattern	

Click Yes on the popup window to save your pattern.

50. Drag and Drop RoutingPattern under the newly created "Routing" Pattern category.

Save the pattern.

3. Testing and Debugging the Pattern

1. Select the "Create Pattern" tab and click "Debug Pattern".

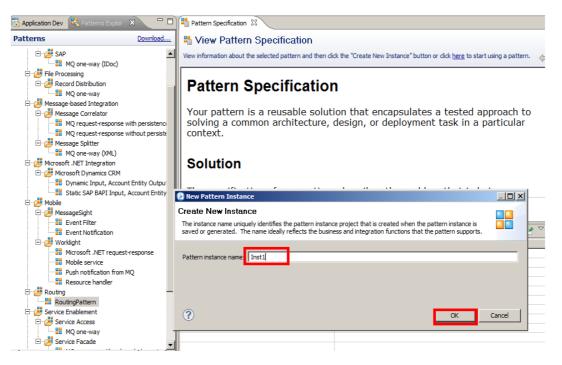
🔠 RoutingFlow.msg	flow 🔡 RoutingPatte	rn.pattern 🔀			
^j Create Pat	tern				
Test your pattern by	v configuring your pattern p	lug-in information,	click "Test Pattern".		
Create	Pattern Plug-ins	0	Test Pattern		🐝 Debug Pattern
Plug-in Information	DN				
🛃 Configure the	unique identifier for your p	attern plug-in.			
Pattern name:	RoutingPattern				
Plug-in ID:	com.your.comp	any.domain.Routi	ngPatternProject		
Version:	1.0.0.0				
Provider:	Your Company	Name			
Description:	Plug-in created	by the Pattern Au	thoring editor		
Translation Optio	ns				
	option, the Pattern Authorii 3 a single language pattern,			-ins. These plug-	ins are set up so that you can dro
Create transla	tion plug-ins (*.nl1 and *.do	oc.nl1)			
Pattern Distributi	on				
After you have te	sted your pattern plug-ins, j	package them into	a pattern archive by c	licking "Create P	attern Archive" and then distribut
•					
Source Files Pattern	Configuration Categories	Create Pattern			

2. Accept the default workspace and click OK.

3. In the new Toolkit instance, click on the "Patterns Explorer" tab.

Double-click on "RoutingPattern" to create a new instance.

Enter "Inst1" as the Pattern instance name and click OK.



4. In the Pattern Configuration, expand the "General Parameters" group.

Enter "RoutingPatternProject" as the Project Name.

Battern Specification	🚦 *Inst1 - Pattern Configuration 🛛		
🌯 Configure Patte	rn Parameters		
Provide values for pattern para	ameters. Click the "Generate" button or click	<u>here</u> to generate a pattern instance.	
🔕 The pattern parameter "Rou	uting Table" is mandatory but a value is not s	et.	
Pattern Parameters			E E 🖬
▼ General Parameters			8
General Parameters			
Project Name *	RoutingPatternProject		
OutputsAmount *	1		~
Routing Table *	Filter Pattern	Output Terminal	Add
			Edit
			Delete
	<		<u>></u> 2 3
MQOutput1			8

Note that since OutputsAmount is set to "1", there is only one MQOutput group (MQOutput1)

- General Parameters 0 General Parameters Project Name * RoutingPatternProject OutputsAmount * 5 ~ Routing Table * Filter Pattern Output Terminal Add... Edit. Delete 2 3 > < MQOutput1 8 MQOutput2 8 MQOutput3 8 MQOutput4 0 MQOutput5 8
- 5. Change OutputsAmount to "5" and check that 5 MQOutput groups are shown.

6. Change OutputsAmount to "3" and insert the following 3 rows in the Routing Table (using the Add button):

eneral Parameters				
Project Name *	Routing	PatternProject		
OutputsAmount *	3			~
Routing Table *	Fil	ter Pattern	Output Terminal	Add
Routing Table *	\$R	oot/MQMD/PutApplType = 1	Output Terminal CICS	
Routing Table *	\$R \$R	oot/MQMD/PutApplType = 1 oot/MQMD/PutApplType = 9		Add Edit
Routing Table *	\$R \$R	oot/MQMD/PutApplType = 1	CICS	

Note – if you specify OutputsAmount to a larger number, make sure you create the corresponding number of Routing Tables entries the java code is not production-ready....

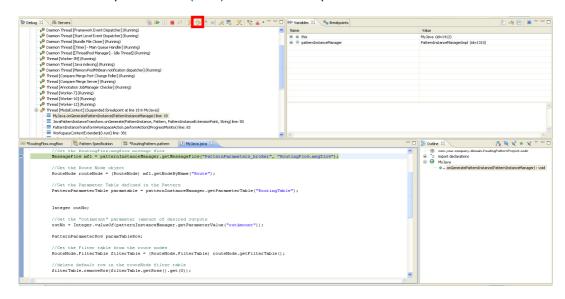
7. Now complete the Queue Name and Queue Manager parameters for the 3 MQOutput groups:

MQOutput1 4QOutput1		
Queue Name *	PATTERNS.CICS	
Queue Manager *	MB8QMGR	
MQOutput2 1QOutput2		
Queue Name *	PATTERNS.WINDOWS	
Queue Manager *	MB8QMGR	

- 8. Click the Generate button.
- 9. Since you put a breakpoint in MyRoutingJava.java, the debugger will pause the execution of the java code and allow you to run it step by step.

Go back to the original Toolkit instance, and click Yes in the "Confirm Perspective Switch" popup to switch to the Debug perspective.

10. Click on the "Step Over" icon () to move one step forward.



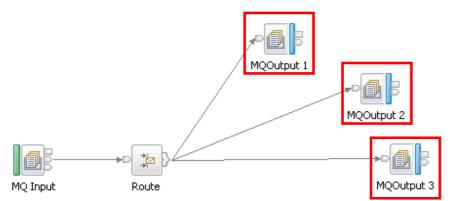
You can go through the code step by step, or click on the "Resume" icon (¹) and let it run until the end.

11. When finished, go back to the second (Test) Toolkit instance, and expand the generated Application Inst1_RoutingPatternProject.

Note that the Application name is composed by the Instance name as a prefix (Inst1), and the Project Name parameter you entered in the Pattern Configuration Editor (RoutingPatternProject).

Double-click on the RoutingFlow.msgflow to open it.

12. Check that the message flow has 3 MQOutput Nodes, each with its defined Queue Name and Queue Manager properties.



13. Double-click on the Route node and verify that its Filter table was completed using the Routing Table parameter:

Properties	🕄 🔣 Problems 🗄	Deployment Log		1
🗖 Route N	lode Properties - R	oute		
Description	-			
Basic	Filter table*	Filter pattern	Routing output terminal	Add
lonitoring		\$Root/MQMD/PutAppIType = 1 \$Root/MQMD/PutAppIType= 9	CICS WINDOWS	Edit
		\$Root/MQMD/PutApplType = 28	JAVA	Delete
				(名)
	Distribution mode	Al		

This concludes the "Groups and Table Parameters" Pattern Authoring lab.