

Cognizant Technology Solutions

Exercises – Basic Flow Steps

WebMethods Integration Workshop

WebMethods CoE

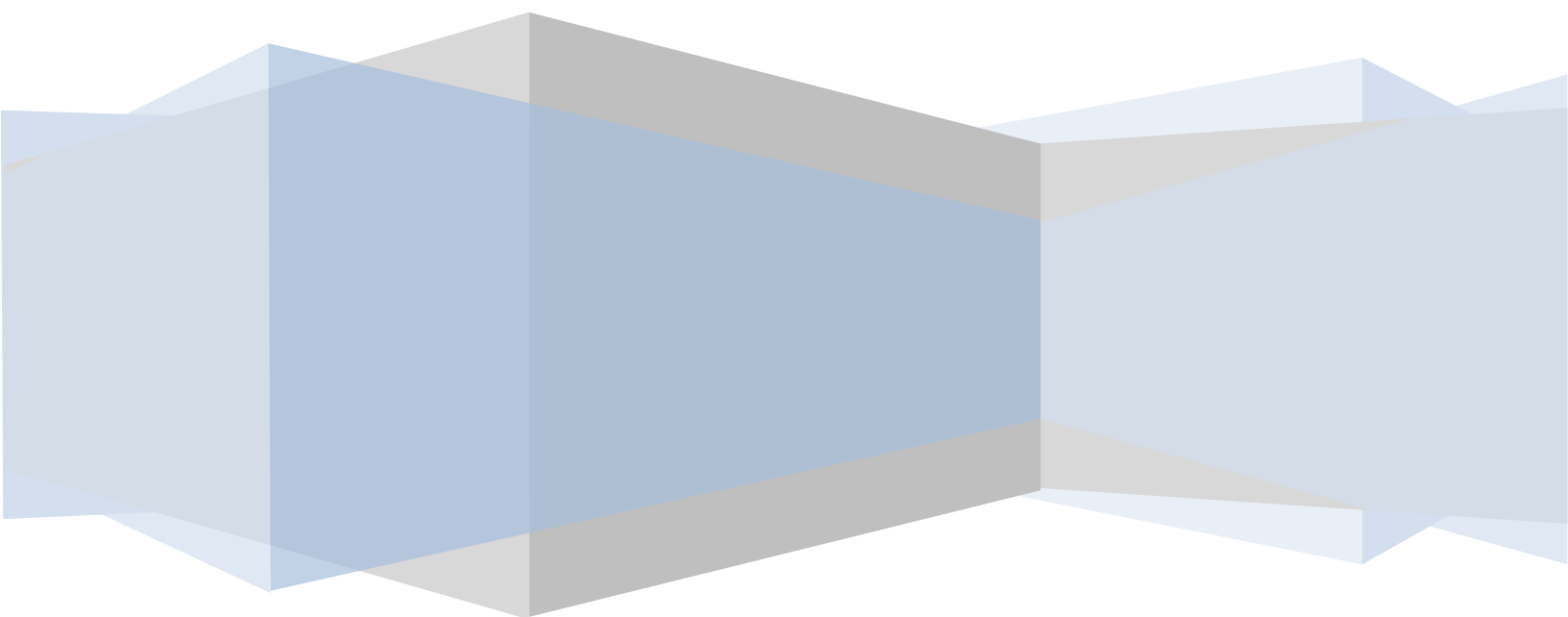


Table of Contents

Introduction	3
Map.....	3
Invoke	3
Document Type	4
Branch.....	4
Theory.....	4
Practical	5
INVOKE Exercises.....	5
Steps to create INVOKE Flow step:	5
Document Type Exercise.....	7
Task	7
Steps to create a document:	8
BRANCH Exercise.....	11
Task	11
Steps to create BRANCH Flow step:.....	12

Flow Step

Introduction

A flow service contains **flow steps**. A flow step is a basic unit of work (expressed in the webMethods flow language) that webMethods Integration Server interprets and executes at run time. The webMethods flow language provides flow steps that invoke services and flow steps that let you edit data in the pipeline.

webMethods' flow language also provides a set of control steps that allow you to direct the execution of a flow service at run time. The control steps allow you to:

- Conditionally execute a specified sequence based on a field value.
- Retry a specified sequence until it succeeds.
- Repeat a specified sequence (loop) for each element in an array field.

Map

Basic mapping tasks are the tasks you perform to manage the pipeline contents and the values of variables in the pipeline.

In the Pipeline view, you can perform the following basic mapping tasks:

- Link variables to each other. You can copy the value of a variable in one service or document format to a variable in another service or document format.
- Assign values to variables. You can hard code variable values or assign a default value to variables.
- Drop variables from the pipeline. You can remove pipeline variables that are not used by subsequent services in a flow.
- Add variables to the pipeline. You can add variables that were not declared as input or output parameters of the flow service. You can also add input and output variables for services that the flow service invokes (internally invoked services).

Invoke

Use the INVOKE step to request a service within a flow. You can use the INVOKE step to:

- Invoke *any* type of service, including other flow services and Web service connectors.
- Invoke any service for which the caller of the current flow has access rights on the local webMethods Integration Server.
- Invoke built-in services and services on other webMethods Integration Servers.
- Invoke flow services recursively (that is, a flow service that calls itself). If you use a flow service recursively, bear in mind that you must provide a means to end the recursion.

Document Type

Documents are objects that webMethods components use to encapsulate and exchange data. A document represents the body of data that a resource passes to webMethods components. An IS document type contains a set of fields used to define the structure and type of data in a document (IData object). Often it represents a business event such as placing an order (purchase order document), shipping goods (shipping notice), or adding a new employee (new employee record).

Branch

The BRANCH step allows you to conditionally execute a step based on the value of a variable at run time. For example, you might use a BRANCH step to process a purchase order one way if the **PaymentType** value is “**CREDIT CARD**” and another way if it is “**CORP ACCT**”.

When you build a BRANCH step, you can:

- **Branch on a switch value.** Use a variable to determine which child step executes. At run time, the BRANCH step matches the value of the **switch** variable to the **Label** property of each of its targets. It executes the child step whose label matches the value of the switch.
- **Branch on an expression.** Use an expression to determine which child step executes. At run time, the BRANCH step evaluates the **expression** in the **Label** property of each child step. It executes the first child step whose expression evaluates to “**true**.”


Theory

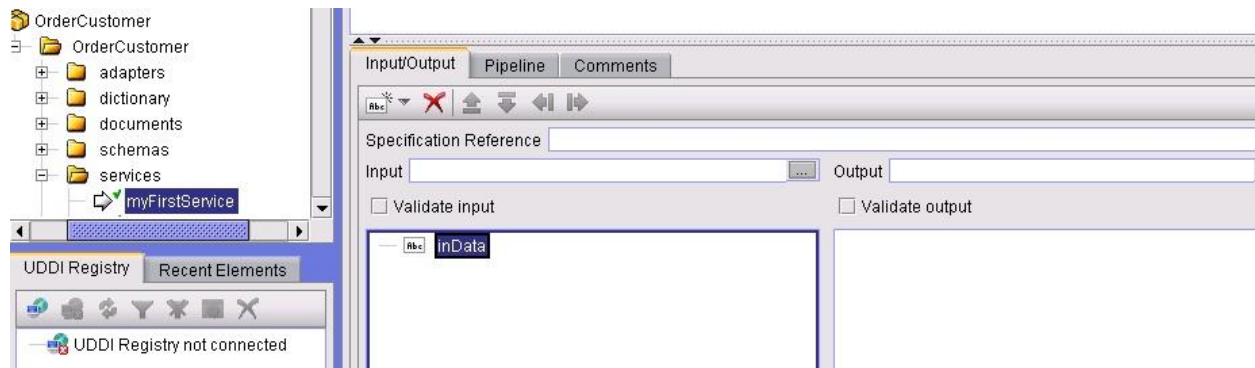
- 8-2-SP1_Developer_Users_Guide (Chapter 6, Chapter 7 (MAP, INVOKE & BRANCH))
- 8-2-SP1_Service_Development_Help (Chapter 8, Chapter 9)


Practical

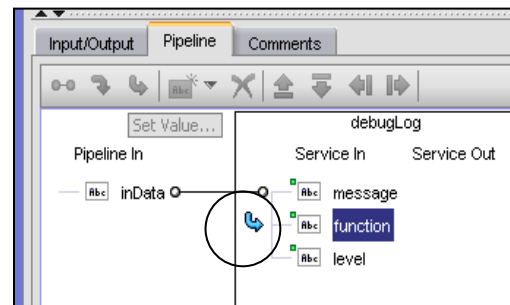
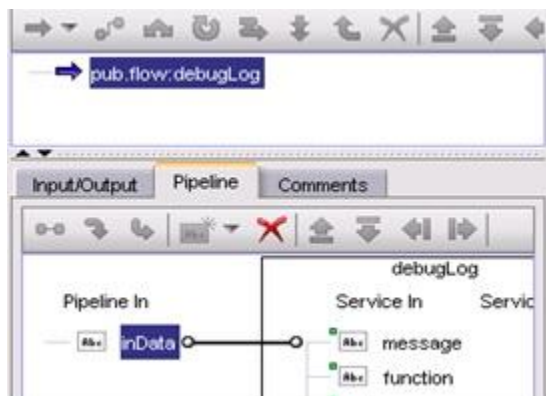
INVOKE Exercises

Steps to create INVOKE Flow step:


- 1) From the menu bar choose **File | New | Flow Service**. Select the appropriate folder (**services**) and type the **name** of the Service "**myFirstService**". **Finish**.
 - Click once in the white space in the input pane (bottom middle panel) of the Input/Output tab (to establish focus) and click the **New Variable** icon . Select the **String** data type and name the new input string field "**inData**". You should now see the following:



- Click once in the white space of the edit panel (top middle panel) and click the **Insert** button, , **Browse | WmPublic | pub | flow | debugLog**. **OK**.
- In the middle lower pane, select the Pipeline tab. Map a field by selecting it and without releasing the mouse button, dragging a line to the target field. Map the Pipeline In **inData** field to the Service In **message** field. Select the **function** field and use the **Set Value...** button to hard-code your name. Press OK.

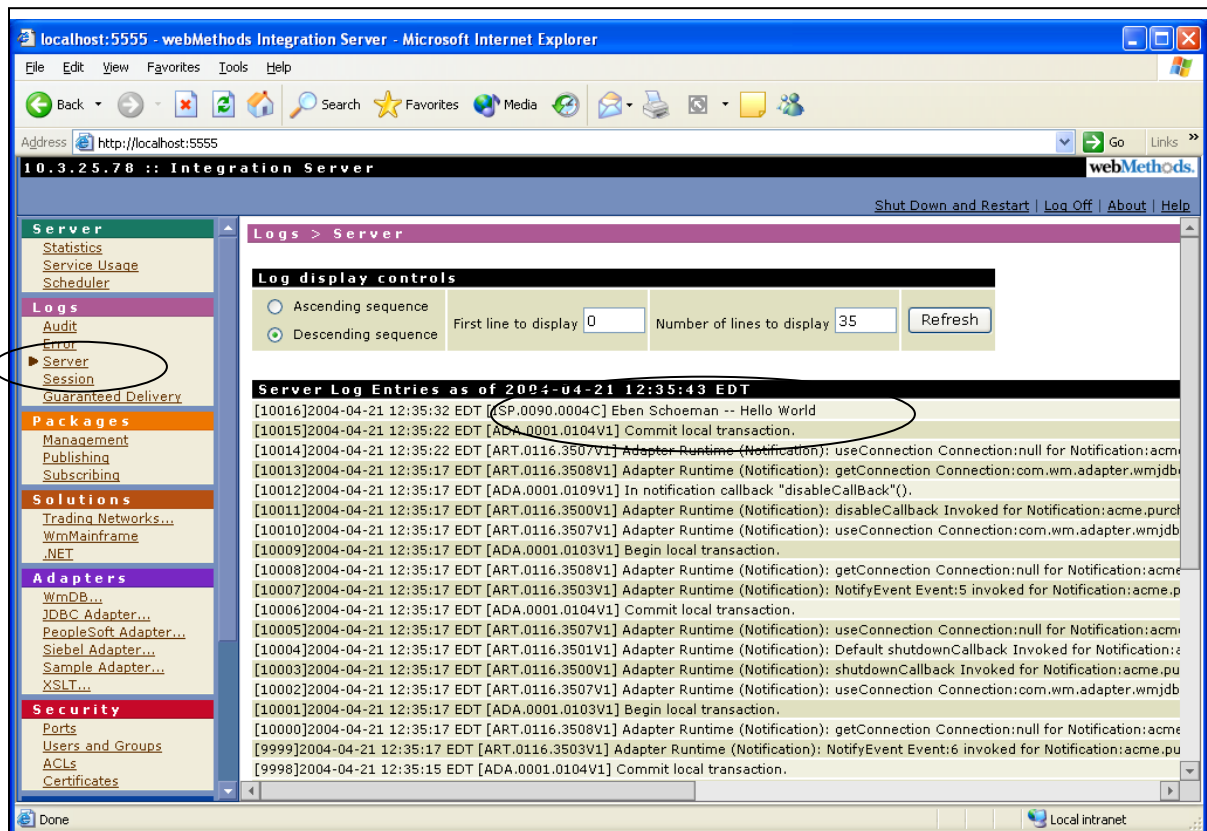


- Save your service with either **File | Save**, or click on the **Save** icon. .

2) Run the service. Either choose **Test | Run** from your menu bar, or click on the **Run** icon . Provide the *inData* input string a value of "**Hello World**". **OK**

- Confirm successful execution in the server log. Developer **Results** panel, and also in the **Server Log** (DOS window if the IS was started from the command prompt with a -log none, or look at the **Server Administration** tool, **Logs | Server**). You should see the following results:

Results		
Name		Value
Abc	message	Hello World
Abc	function	Eben Schoeman
Abc	inData	Hello World



The screenshot shows the webMethods Integration Server interface in a Microsoft Internet Explorer browser window. The address bar shows `http://localhost:5555`. The main content area displays the **Logs > Server** section. On the left sidebar, the **Logs** section is expanded, and the **Server** log is selected. The **Log display controls** section shows the **Descending sequence** radio button selected, with the **First line to display** set to 0 and the **Number of lines to display** set to 35. The **Server Log Entries as of 2004-04-21 12:35:43 EDT** section shows a list of log entries. The entry at index 10016, timestamped 2004-04-21 12:35:32 EDT, is circled and shows the message: `[ASP.0090.0004C] Eben Schoeman -- Hello World`. Other entries include transaction commits, adapter runtime notifications, and shutdown callbacks.

Document Type Exercise

Task

Before we can write effective code, we need to agree on data exchange message formats and create the initial document type structures for use in our business case.

- A] One of ORDERCUSTOMER's major customers has already integrated with another supplier using XML over HTTP. Your team has decided to re-use this customer's existing Order Request XML Schema in order to simplify the project. This standard uses Dun & Bradstreet numbers (<http://www.dnb.com>) as trading partner identifiers.
- B] You have also done a review of your back-end systems and must now define a canonical order document type for use internally.
1. Use a text editor to look at the XML document
[DIRHOME]\ORDERCUSTOMER\PORequest.xml. What is the Root node?



PORequest.zip

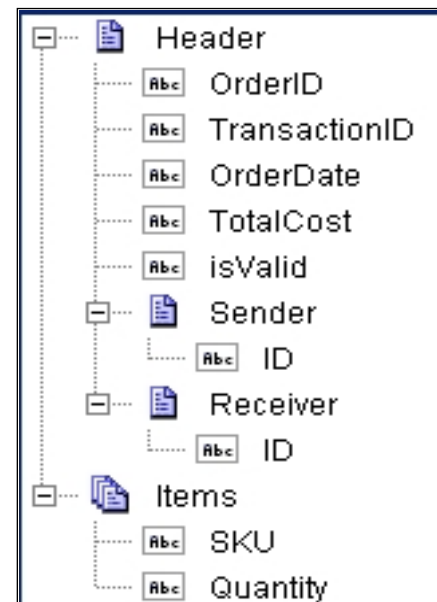
Resources – PORequest.xml attached here -->



OrderRequest.zip

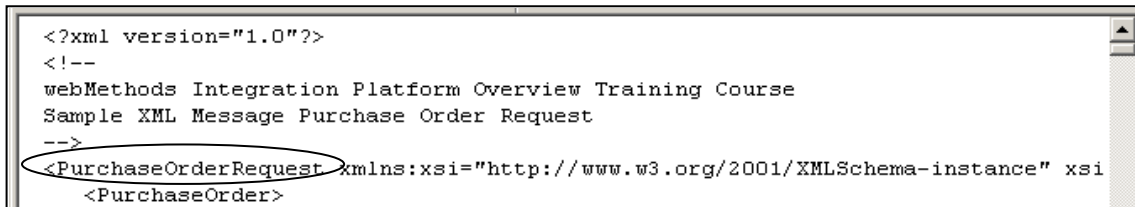
OrderRequest.xsd attached here -->

2. In the **OrderCustomer.OrderCustomer.documents** folder, create the **OrderRequest** Document Type from a provided schema called **<http://xxx:5555/OrderCustomerSupport/OrderRequest.xsd>**. Select the root node you discovered in step 1.
3. In the same **OrderCustomer.OrderCustomer.documents** folder, create the **OrderCanonical** Document Type manually as shown. Test your services by expanding and collapsing both the **Header** document and the **Items** document type.



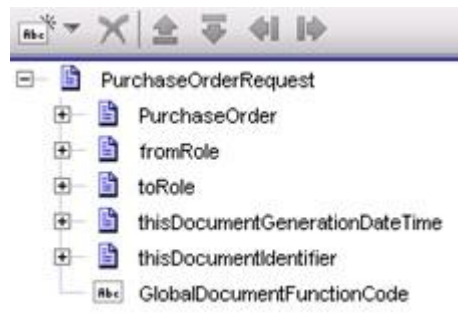
Steps to create a document:

1. Open the XML file, **PORequest.xml**, in a text editor (located in the following directory **DIRHOME\OrderCustomerSupport**) and note the root node of this document: **PurchaseOrderRequest** as shown below:






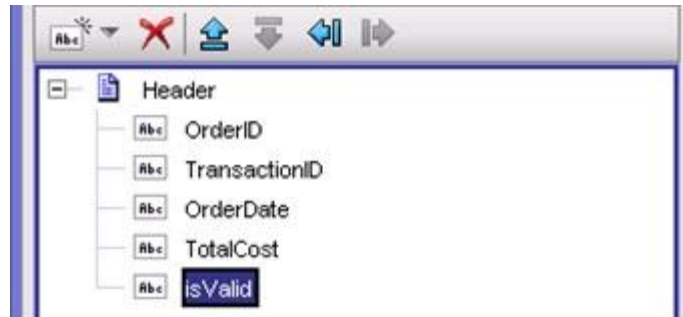
```
<?xml version="1.0"?>
<!--
webMethods Integration Platform Overview Training Course
Sample XML Message Purchase Order Request
-->
<PurchaseOrderRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi
  <PurchaseOrder>
```








2. Select the **documents** folder, **File | New | Document Type | Next**, name is **"OrderRequest"**, **Next | XML Schema | select the http://xxx:5555/OrderCustomerSupport/OrderRequest.xsd file | Next | highlight PurchaseOrderRequest | Finish**. You should see:

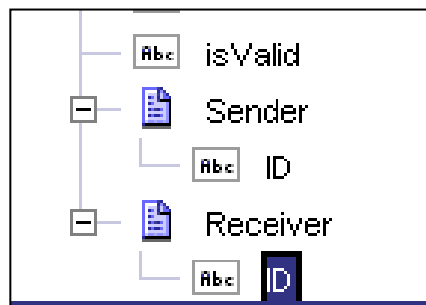







3. To create the manual **OrderCanonical**:

- Select the **documentss** folder, **File | New | Document Type | Next**, name is **"OrderCanonical"**, **Next | None | Finish**.
- Select the **New Variable** icon, , **Document | "Header"**.
- Select the **New Variable** icon, , **String | "OrderID" | <carriage return> key**.
- Indent under Header by clicking on the **Shift Right** icon, .
- Next, add the strings: **"TransactionID, OrderDate, TotalCost, isValid"**. You should not need to indent them. You should see the following:



- Select the **New Variable** icon, , **Document** | **"Sender"**. Select the **New Variable** icon, , **String** | **"ID"** and indent under **Sender** document by clicking on the **Shift Right** icon, .
- Select the **New Variable** icon, , **Document** | **"Receiver"** and shift it left by clicking on the **Shift Left** icon, . Select the **New Variable** icon, , **String** | **"ID"** and indent under **Receiver** document by clicking on the **Shift Right** icon, . You should see:

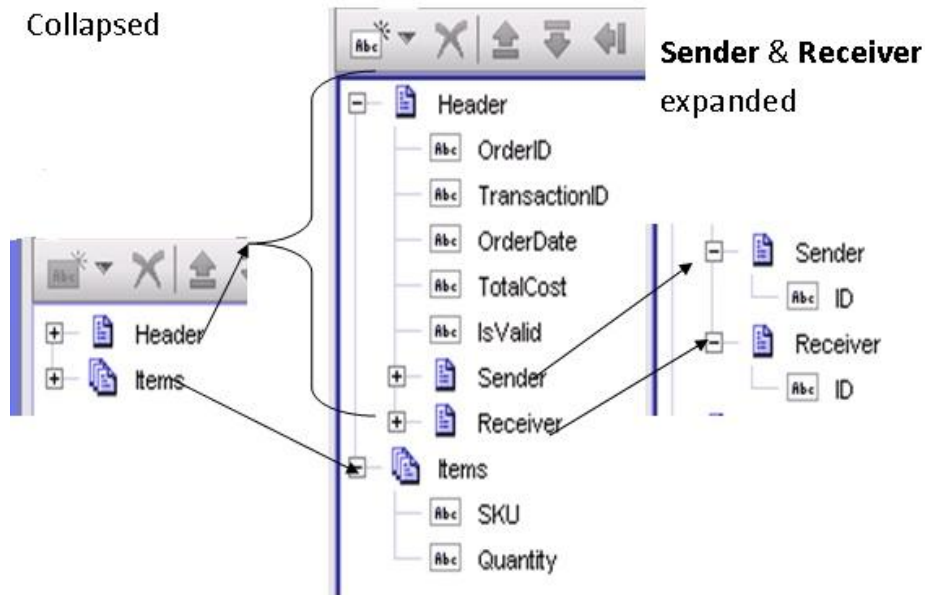


- Select the **New Variable** icon, , **Document List** | **"Items"** and use the **Shift Left** icon, , repeatedly, to move it as far to the left as you can. This will put it at the same level as the Header document.
- Select the **New Variable** icon, , **String** | **"SKU"** and indent under **Items** document list by clicking on the **Shift Right** icon, .
- Select the **New Variable** icon, , **String** | **"Quantity"**.
- Test your services by expanding and collapsing the **Header** document and the **Items** document list. In the Header document, also expand and collapse the Sender and Receiver documents. You should see:

Header & Items

expanded

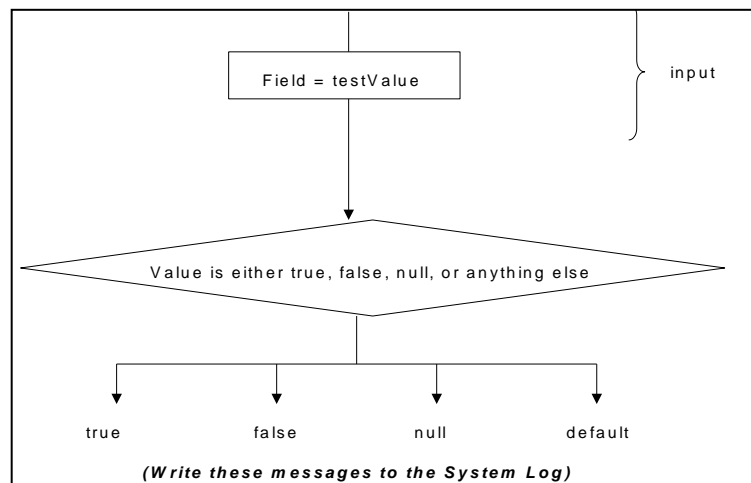
Collapsed



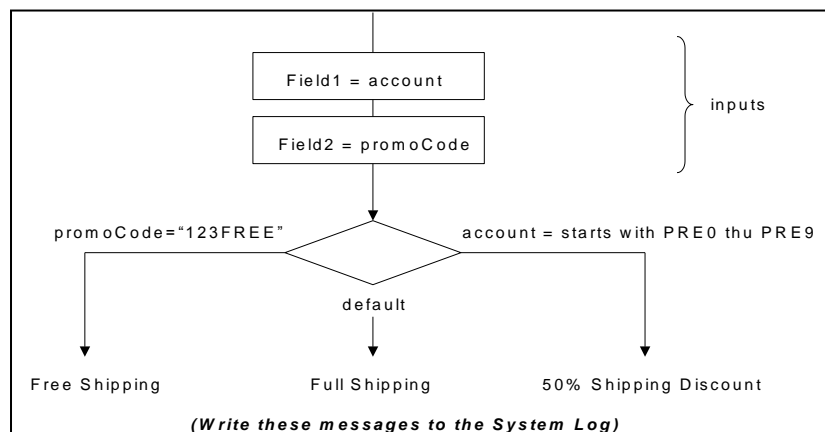
BRANCH Exercise

Task

- 1) In the **service subfolder**, create a service **branch1** with a single input String **"testValue"**. Write **Branch** code to write a message (based on the value of testValue) to the **Server Log**. For example, if **testValue = true** then write, "true" to the server log. See the diagram below. **Test** using the **Step** feature. Output should appear in the **Results** window, or you can look in the server log.



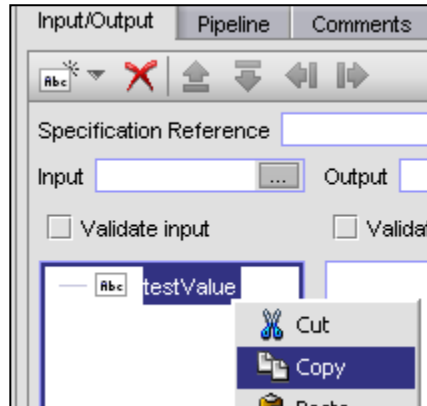
- 2) Create a service **branch2** with two input Strings, **"account"** and **"promoCode"**. Write **Branch** code to write a message (based on the value of the input fields) to the **Server Log**. The psuedocode for this service is as follows:
 - a) If **promoCode = "123FREE"** then write **"Free Shipping"** to the server log.
 - b) Else if **account** starts with PRE0 thru PRE9 then write, **"50% Shipping Discount"** to the server log.
 - c) Else, write **"Full Shipping"** to the server log.
- 3) The diagram below graphically shows the psuedocode for this service. Note that **account = starts with PRE0 thru PRE9** is a conditional expression! **Test** using the **Step** feature.




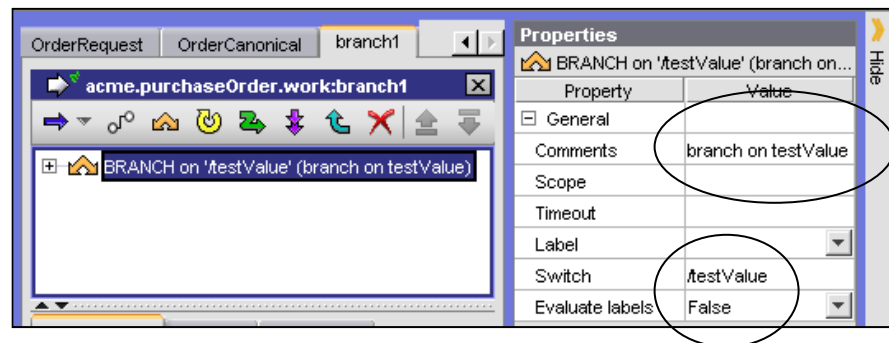
Steps to create BRANCH Flow step:



1) Branch 1

1. Select the appropriate folder (**services**) and create a new **service** structure; name it "**branch1**". **File | New | Flow Service | Next**, select the **services** subfolder and type the name "**branch1**". **Finish**.
- Select the **Input/Output** tab and click once in the **Input** panel white space. Click the "**New variable**" button. Add a **String** and name it **testValue**. Right click **testValue** and choose **Copy**. You will use this value in the next step.



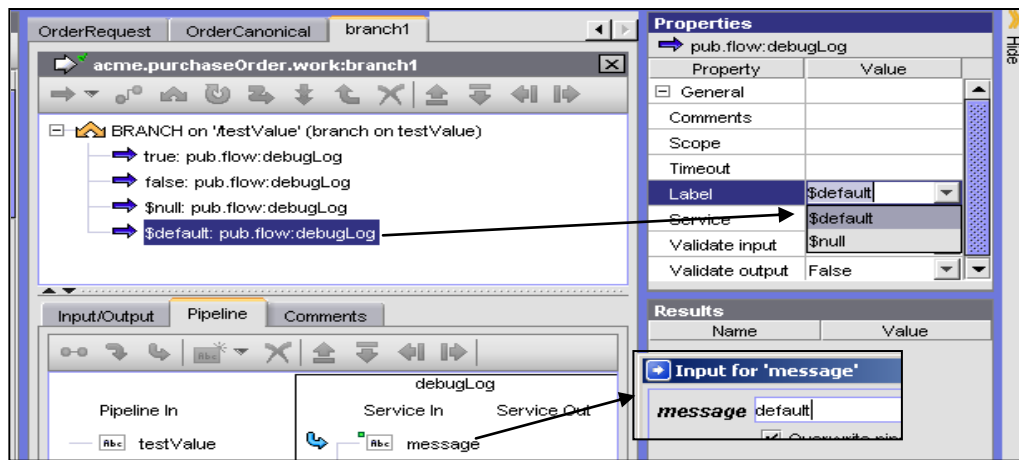
- Click once in the **white space** of the Service (to establish focus, top panel) and click the **Branch** button, . In **Branch Properties**, paste the name of the **Switch** pipeline field – **testValue**. Since you're using the switch value, leave **Evaluate-labels** = "**False**". In the **Branch Properties Comments** field enter: **branch on testValue**





- Click once in the **white space** of the Service (to establish focus, top panel) and click the **Insert** button, , **Browse | WmPublic | pub | flow | debugLog**. **OK**. Indent the **debugLog** built-in Service under the **Branch**, by clicking on the Shift-Right icon, . In the **debugLog Properties**, type a **Label** name = **true**. Select the **Pipeline** tab. You should see the label appear before the **debugLog** statement. In the **Pipeline** tab, double-click **message** and set its value to **true**. Click **OK**.
- Copy the **debugLog** statement and paste in three times into your service. Change the Label of the last three steps from **true** to **false**, **\$null** and **\$default**, respectively. Also double click each **message** in the last three **debugLog**

statements and change them from *true* to **false**, **null** and **default**, respectively. Set the **function** field = <your name>.




- Example:



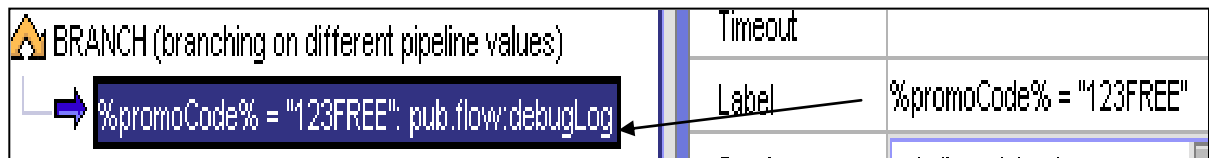
- **To run the service.** **Test | Run** or click on the **Run** icon . Provide the following values and confirm the results.
 - Set *testValue* to **true** and you should see **true** in the server log.
 - Set *testValue* to **false** and you should see **false** in the server log.
 - Set *testValue* to **maybe** and you should see **default** in the server log.
 - Select the value **maybe** and press the **Delete** key to set the *testValue* to *null*, and you should see **null** in the server log.
- To troubleshoot, **Test | Step** or **click on the Step icon**,  to step through the Flow. Note: the Branch with a label of **\$null** means the field is blank! Watch the indentation!

2) Branch 2

2. In the same **services** folder and create a new flow service and **name** it "**branch2**". **File | New | Flow Service | Next**, select the **services** subfolder and type the name "**branch2**". **Finish**.

- Select the **Input/Output** tab, click once in the **Input** panel white space, and click the "**New variable**" button. Add two **Strings** named **account** and **promoCode**.
- **Click once in the white space** of the Service (to establish focus, top panel) and add a Branch step by clicking on the Branch icon, . In Branch properties, leave the **Switch** value blank, set **evaluate-labels** = **True** and set **comment** = **branching on different pipeline values**
 - **Click once in the white space** of the Service (to establish focus, top panel) and click the **Insert** button, , | **pub.flow:debugLog**. **OK**. Indent the **debugLog** built-in Service under the **Branch**, by clicking on the Shift-Right icon, . Select the **Pipeline** tab, and **set** (double-click) **message** = **Free Shipping**. Click **OK**. Copy the variable **promoCode**. Select the **debugLog** statement in the edit panel

(top middle panel). In the debugLog Properties, type the **Label** expression **%% = "123FREE"**. Position your cursor between the two percent signs (%%) and type **<Ctrl-V>** (hold down the **<Ctrl>** key and hit the **V** key). The Label expression should now show **%promoCode% = "123FREE"**. Select the **debugLog** again in the edit panel. You should see the label appear before the debugLog statement as shown below. Set **function** = <your name>:



- b. Copy and paste the previous step. Indent. In the pipeline tab, change the message to **50% Shipping Discount** and copy the **account** variable. Next, select the second **debugLog** and change the **Label** field to **%% = /^PRE[0-9]+/**. Between the percent signs, paste the **account** variable to change the **Label** to: **%account% = /^PRE[0-9]+/**. This regular pattern means:


/^PRE[0-9]+/ Evaluates to true if the value starts with **PRE**, followed by one or more digits

^ - match the beginning of a string or line

[] - match any character within the brackets

+ - match the preceding item one or more times

- c. Copy and paste the previous **debugLog** statement. Indent. Set the message to **Full Shipping** and the **Label** to **\$default**.

3) To run the service. **Test | Run** or click on the **Run** icon . Provide the following values for **account** and **promoCode** input strings. **OK**.

- Set **account** to **PRE8** and **promoCode** to **123FREE**, you should see **"Free Shipping"**
- Change **promoCode** to **None**, you should see **"50% Shipping Discount"**.
- Change **account** to **ABC**, you should see **"Full Shipping"**