MIRACLE SOFTWARE SYSTEMS,INC.

SAP CONNECTIVITY

Table of Contents

Table of Contents..2

1.Java Compute Node

1.1.Overview

1.2.What this lab is about

In this Lab, You will know how to make use of JavaCompute node for reading file contents.

We will use JavaCompute node to create and build a new output message that is totally independent of the input message.

1.3.What you will do in this lab

You will go through each step in the following sections. Each step might contain many subsets

Key Steps are
1) How to Create and Configure JavaCompute Node
2) How to Build and Deploy Java Code used in message flow.

2.Requirements, Prerequisites, and Planning

Software Requirements	Network Requirements	Additional Requirements
All necessary software has been installed on the system.	None	None

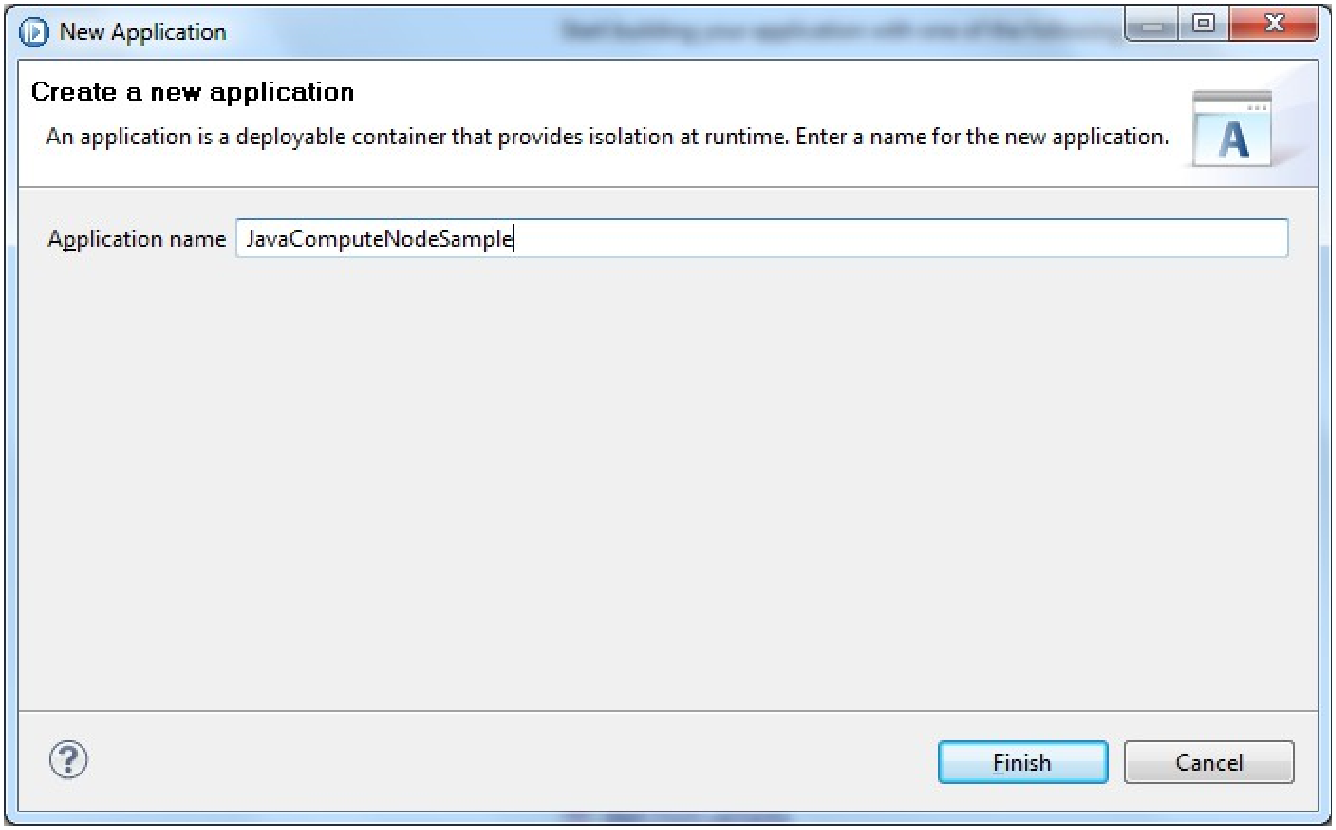
Exercise

The steps involved are

Creating a Flow Using MQInpoutNode, FileOutputNode and JavaCompute node
Configuring a JavaCompute node
Writing java code to read file in JavaComputeNode
Test the flow

3.Scenario:

Step 1:- Create a Application "JavaComputeNodeSample"

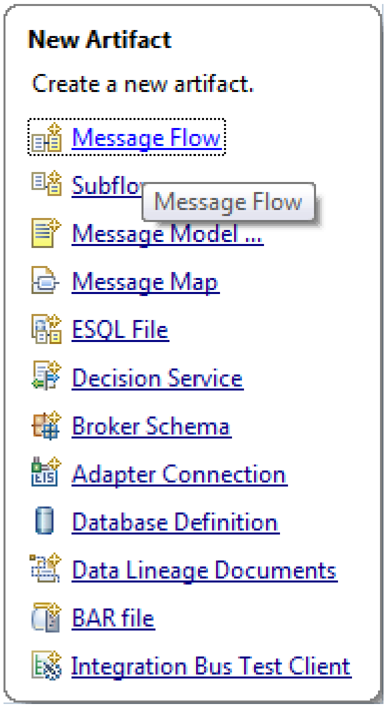


Create a new application

An application is a deployable container that provides isolation at runtime. Enter a name for the new application.

Application name: JavaComputeNodeSample

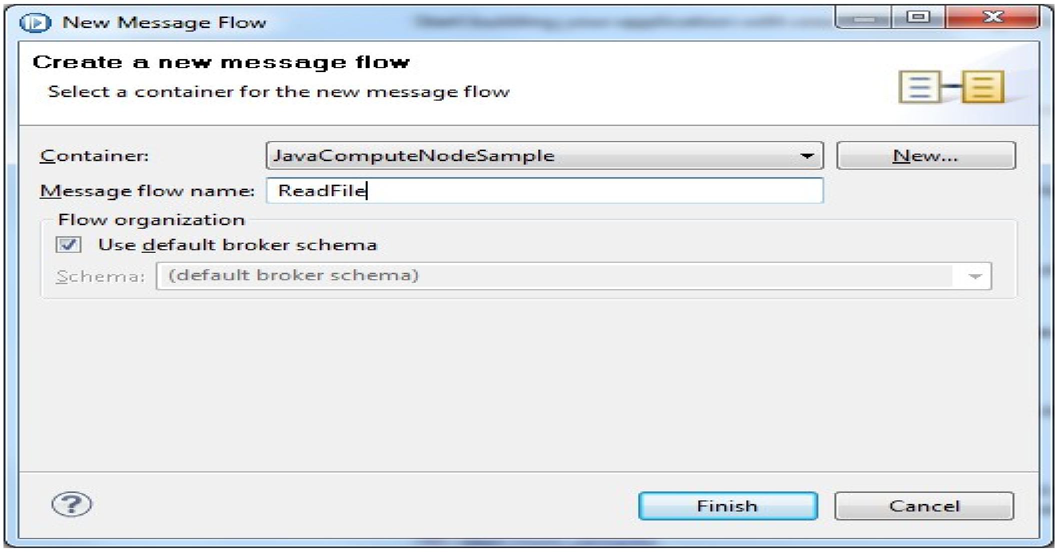
Step 2:- Create new message Flow “ReadFile”



New Artifact

Create a new artifact.

- Message Flow
- Subflow
- Message Model ...
- Message Map
- ESQL File
- Decision Service
- Broker Schema
- Adapter Connection
- Database Definition
- Data Lineage Documents
- BAR file
- Integration Bus Test Client



New Message Flow

Create a new message flow

Select a container for the new message flow

Container: JavaComputeNodeSample

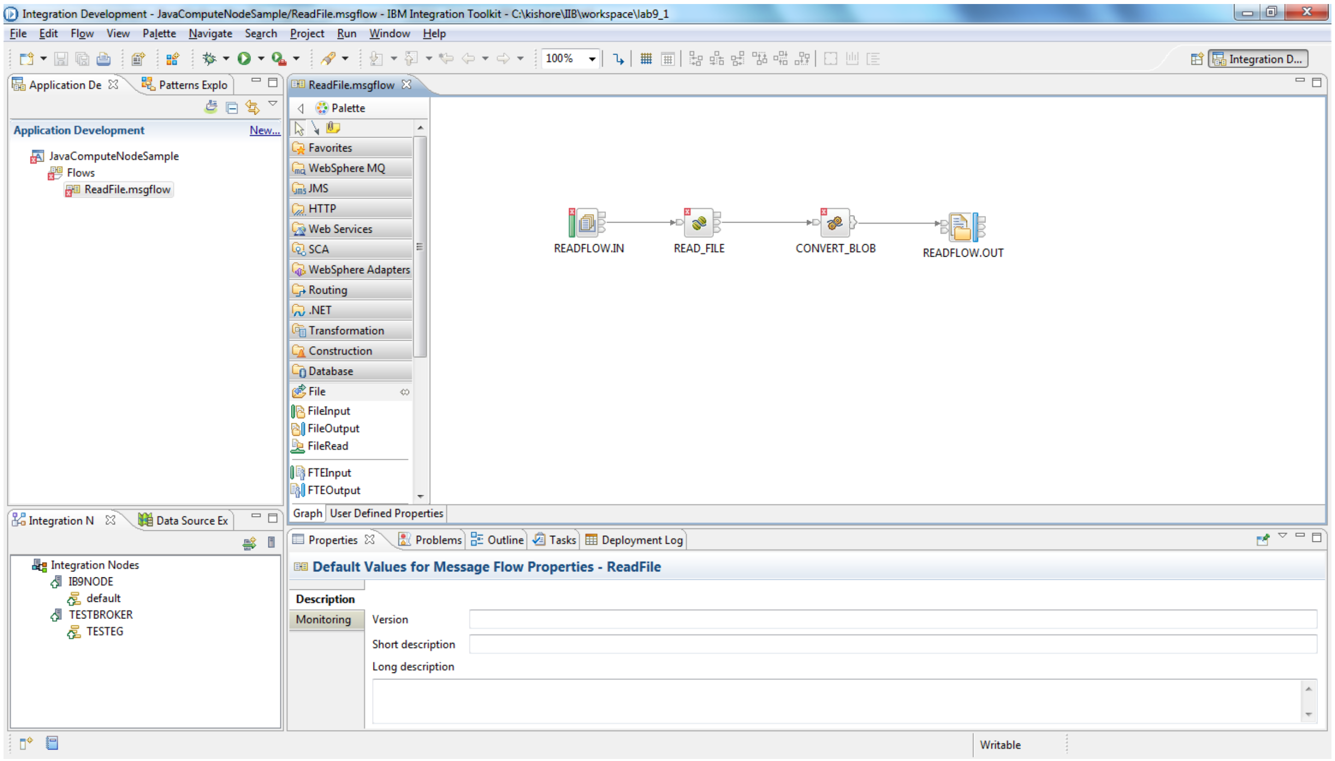
Message flow name: ReadFile

Flow organization

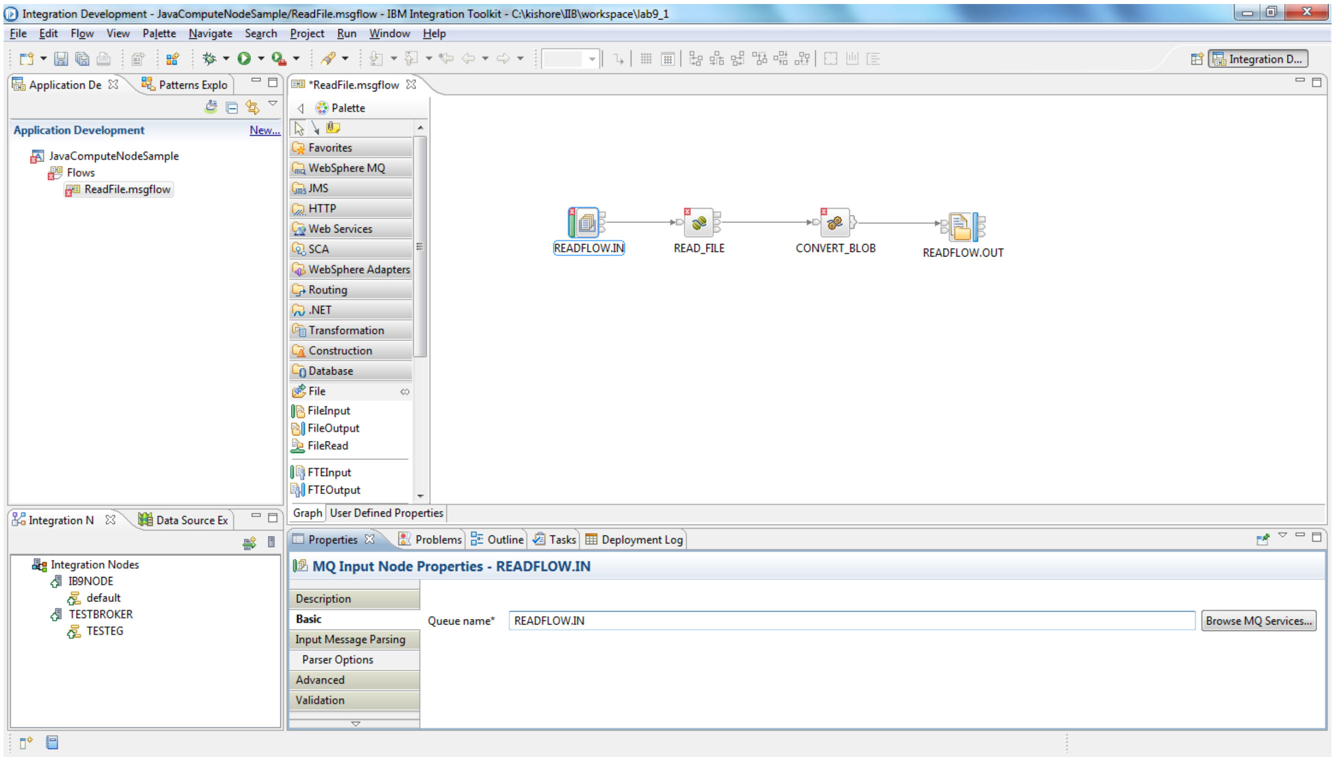
☑ Use default broker schema

Schema: (default broker schema)

Step 3:- Create the flow using MQInputNode, JavaComputeNode,ComputeNode,FileOutputNode as shown in below.



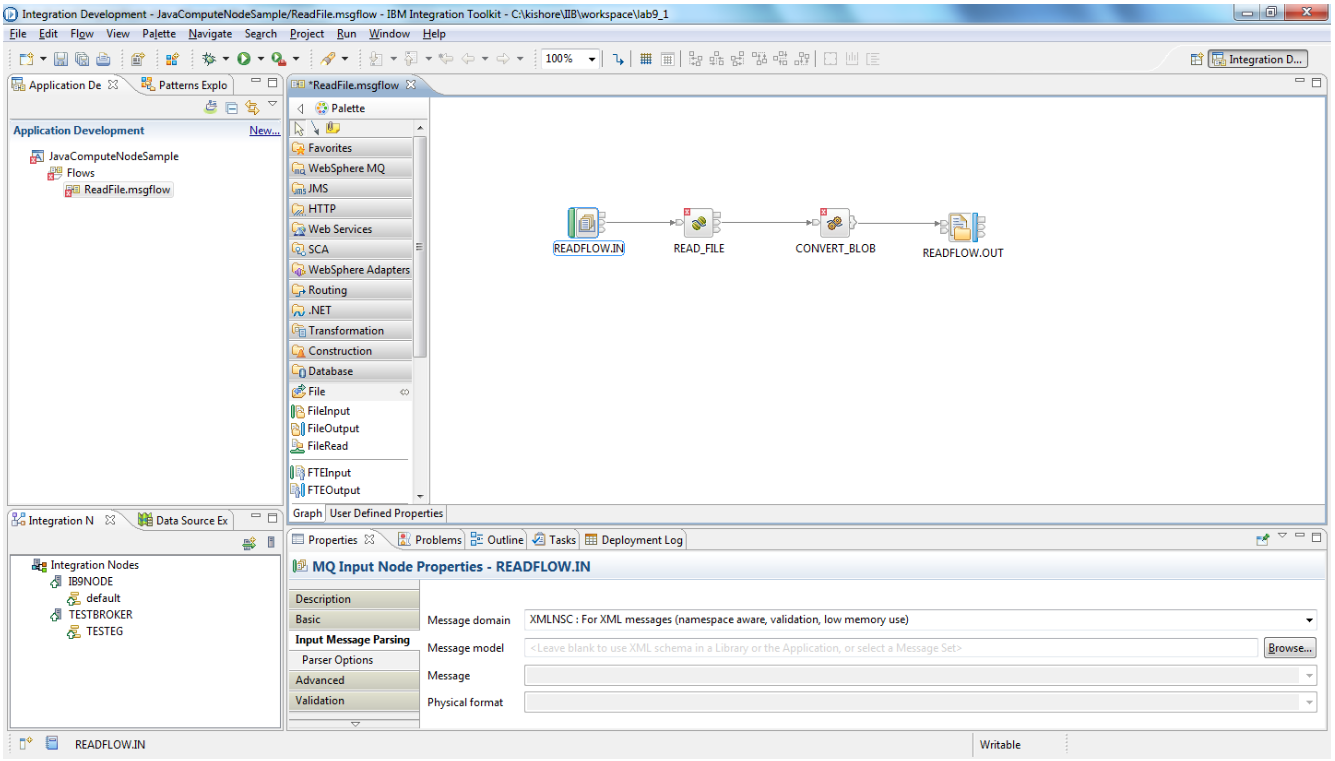
Step 4:- Right click on the MQInputNode and provide the Queue name “READFLOW.IN”



MQ Input Node Properties - READFLOW.IN

Queue name* READFLOW.IN

Select Input message parsing message Domain as XMLNSC



MQ Input Node Properties - READFLOW.IN

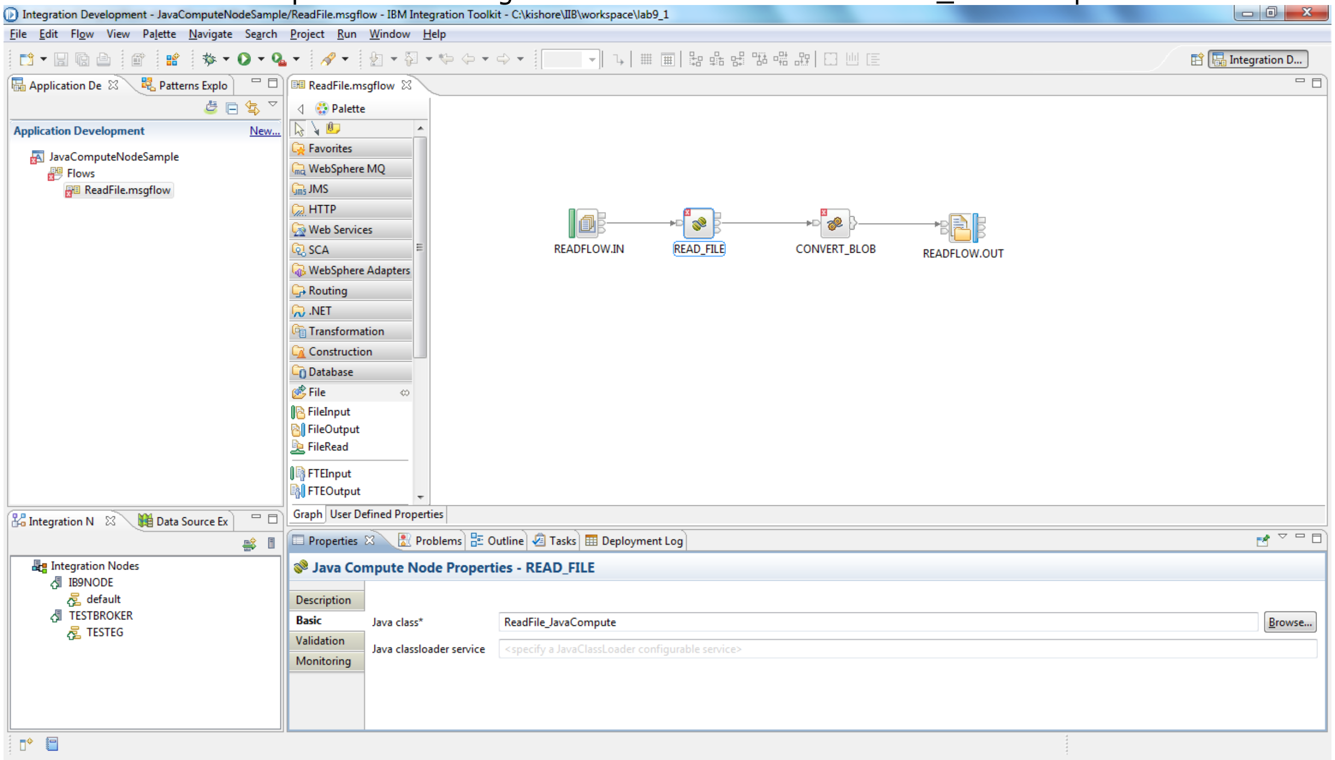
Message domain XMLNSC : For XML messages (namespace aware, validation, low memory use)

Message model <Leave blank to use XML schema in a Library or the Application, or select a Message Set>

Message

Physical format

Step 5:- Configuring the JavaCompute Node.
Click on the JavaCompute Node and give the class name as “ReadFile_JavaCompute”



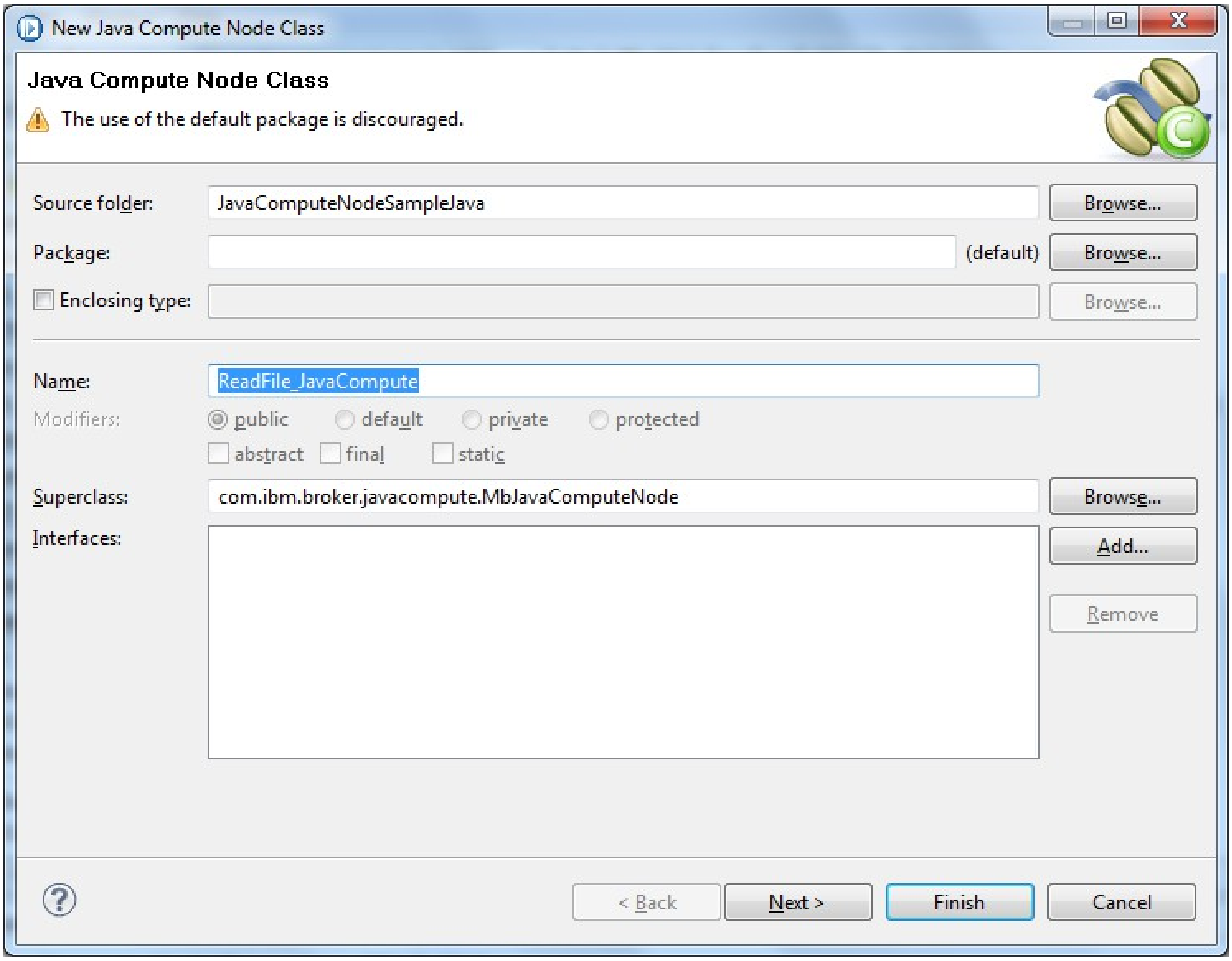
Integration Development - JavaComputeNodeSample/ReadFile.msgflow - IBM Integration Toolkit - C:\kishore\IIB\workspace\lab9_1

Java Compute Node Properties - READ_FILE

Java class* ReadFile_JavaCompute

Java classloader service <specify a JavaClassLoader configurable service>

Double click on the JavaComputeNode. It opens the java class details window. Click Finish.



New Java Compute Node Class

Java Compute Node Class

The use of the default package is discouraged.

Source folder: JavaComputeNodeSampleJava

Package: (default)

□ Enclosing type:

Name: ReadFile_JavaCompute

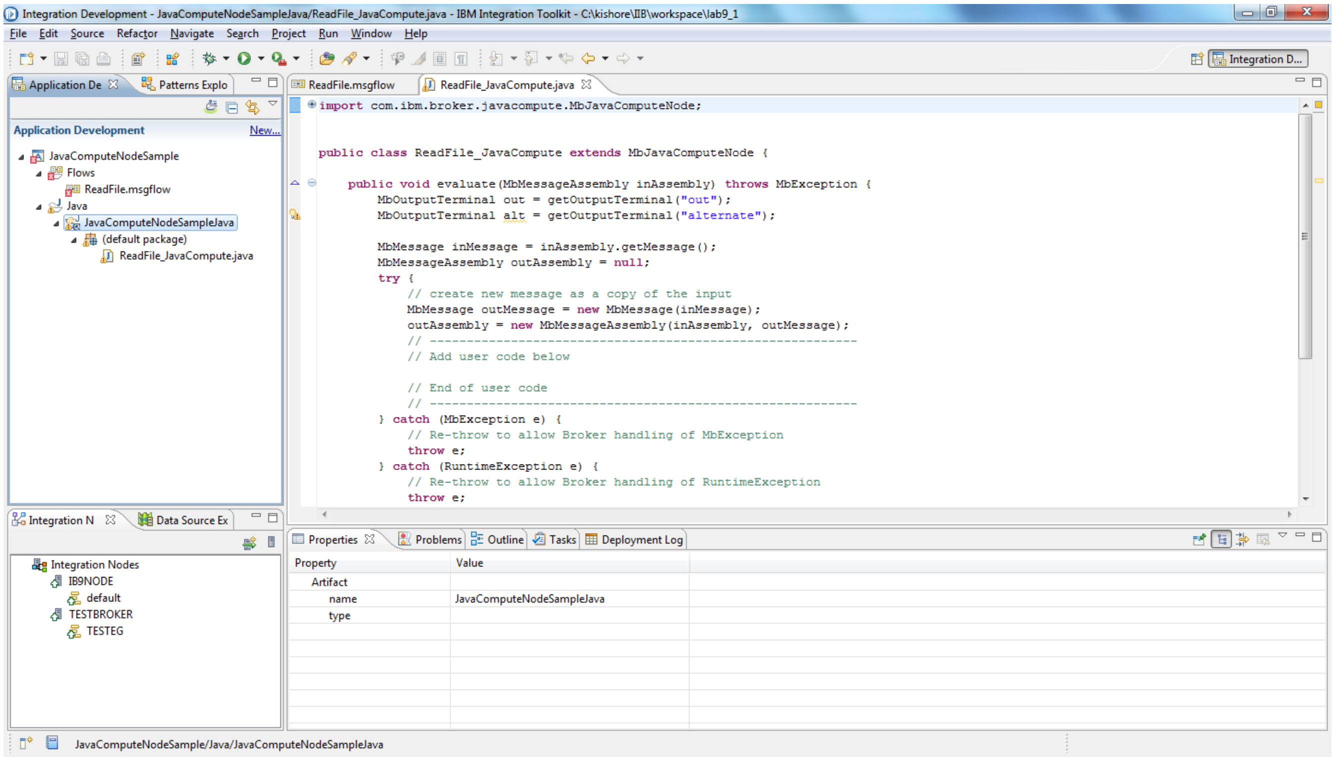
Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: com.ibm.broker.javacompute.MbJavaComputeNode

Interfaces:

It opens the Java class file.


Integration Development - JavaComputeNodeSampleJava/ReadFile_JavaCompute.java - IBM Integration Toolkit - C:\kishore\IIB\workspace\lab9_1
File Edit Source Refactor Navigate Search Project Run Window Help
Integration D...
Application De
Patterns Explo
Application Development
New...
JavaComputeNodeSample
Flows
ReadFile.msgflow
Java
JavaComputeNodeSampleJava
(default package)
ReadFile_JavaCompute.java
ReadFile.msgflow
ReadFile_JavaCompute.java
Integration N
Data Source Ex
Integration Nodes
IB9NODE
default
TESTBROKER
TESTEG
Properties
Problems
Outline
Tasks
Deployment Log
JavaComputeNodeSample/Java/JavaComputeNodeSampleJava

```
import com.ibm.broker.javacompute.MbJavaComputeNode;

public class ReadFile_JavaCompute extends MbJavaComputeNode {

    public void evaluate(MbMessageAssembly inAssembly) throws MbException {
        MbOutputTerminal out = getOutputTerminal("out");
        MbOutputTerminal alt = getOutputTerminal("alternate");

        MbMessage inMessage = inAssembly.getMessage();
        MbMessageAssembly outAssembly = null;
        try {
            // create new message as a copy of the input
            MbMessage outMessage = new MbMessage(inMessage);
            outAssembly = new MbMessageAssembly(inAssembly, outMessage);
            // ----------------------------------------------------------
            // Add user code below

            // End of user code
            // ----------------------------------------------------------
        } catch (MbException e) {
            // Re-throw to allow Broker handling of MbException
            throw e;
        } catch (RuntimeException e) {
            // Re-throw to allow Broker handling of RuntimeException
            throw e;
```

Property	Value
Artifact	
name	JavaComputeNodeSampleJava
type	

Copy the below code in between // Add user code and // End user code comments

```
MbElement mbElement = outAssembly.getMessage().getRootElement();
                    setAttribute("firstParserClassName", "XMLNSC");
                    MbElement mbElement1 = mbElement.getFirstElementByPath("/XMLNSC/File-
Structure");
                    MbElement mbElement2 = mbElement.getFirstElementByPath("/XMLNSC/File-
Structure/FileName");
                    String fileName = (String)mbElement2.getValue();
                    String data = null;
                    File file1=new File(fileName);
                    data=readTheFile(file1);
                    Object obj = mbElement1.createElementAsLastChild(0x1000000, "FileContent",
data);
```

Copy the below method to the java class

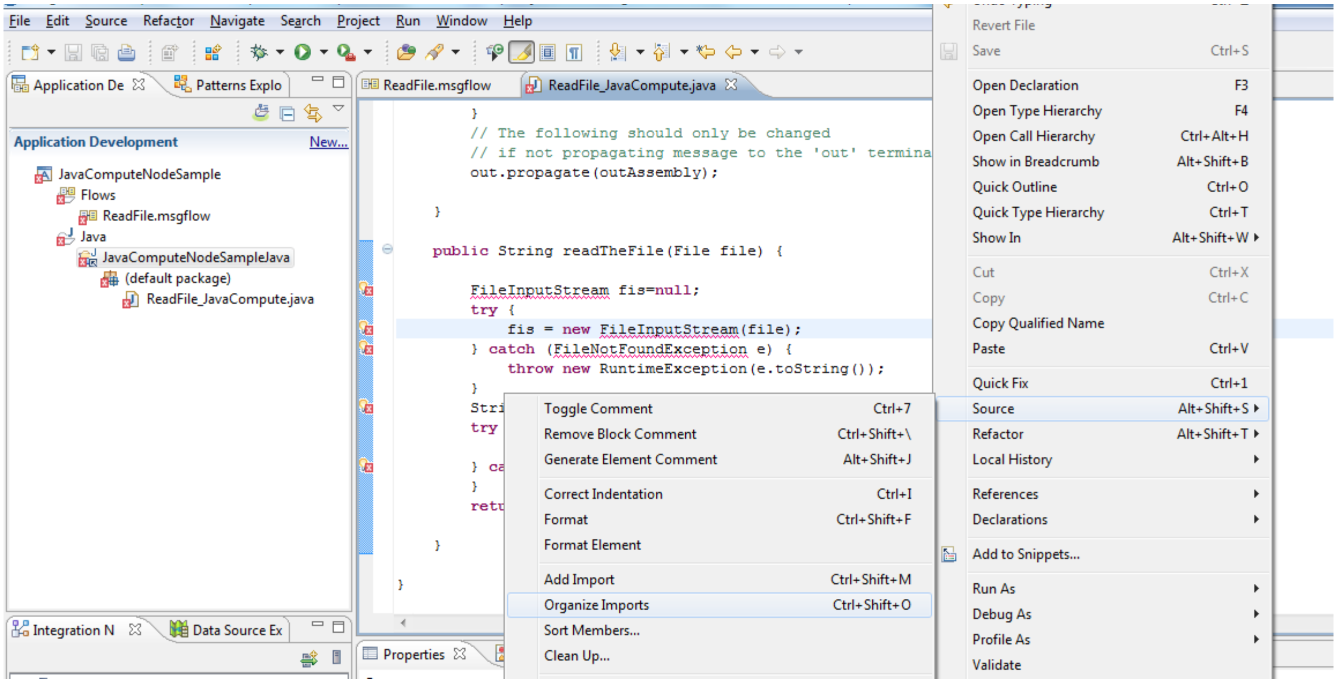
```
        public String readTheFile(File file) {

                FileInputStream fis=null;
                try {
                        fis = new FileInputStream(file);
                } catch (FileNotFoundException e) {
                        throw new RuntimeException(e.toString());
                }
                String inputStreamString = new Scanner(fis,"UTF-8").useDelimiter("\\A").next();
                try {
```

```
                fis.close();
        } catch (IOException e) {
                throw new RuntimeException(e.toString());
        }
        return inputStreamString;

}
```

Right click on the source code and organize imports



```
        }
        // The following should only be changed
        // if not propagating message to the 'out' termina
        out.propagate(outAssembly);

    }

    public String readTheFile(File file) {

        FileInputStream fis=null;
        try {
            fis = new FileInputStream(file);
        } catch (FileNotFoundException e) {
            throw new RuntimeException(e.toString());
        }
        Stri
        try

        } ca
        }
        retu

    }

}
```

Step 6:- Double click on the compute node and copy the below ESQL

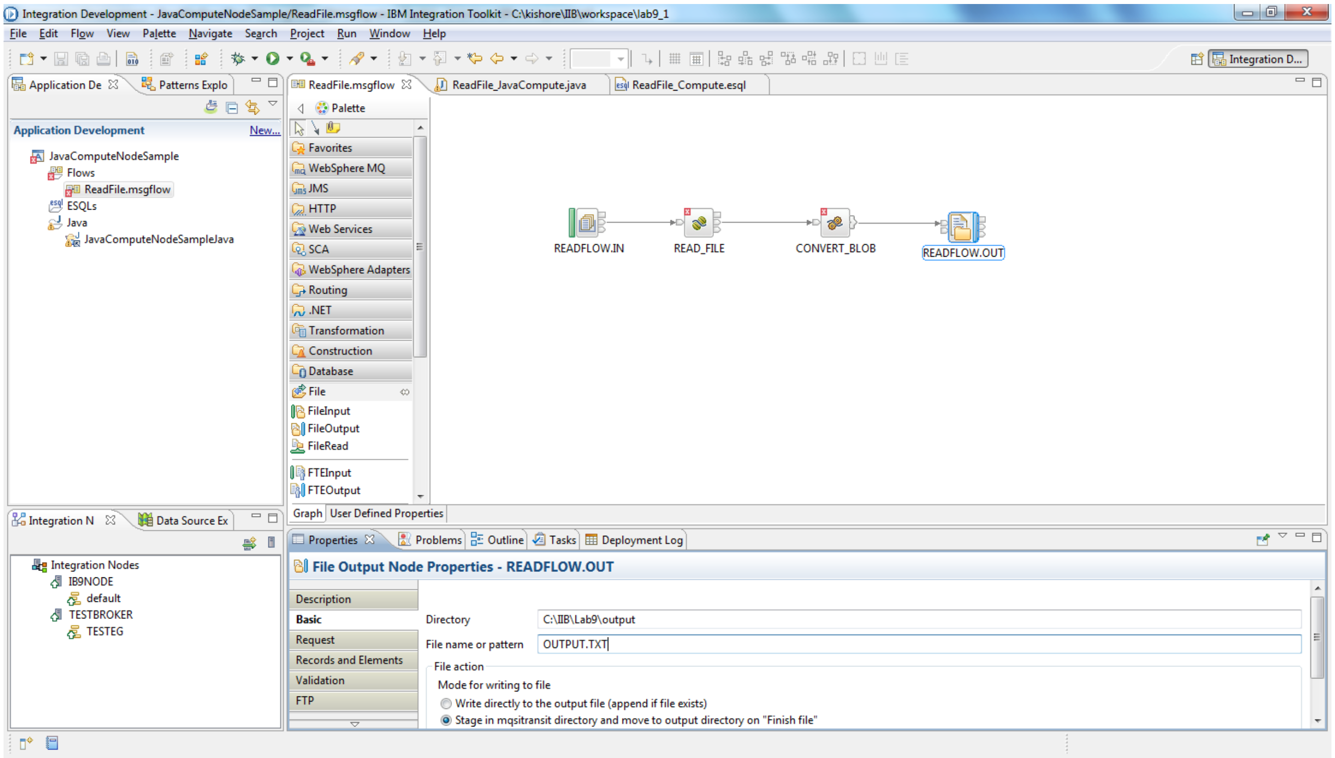
```
CREATE COMPUTE MODULE ReadFile_Compute
        CREATE FUNCTION Main() RETURNS BOOLEAN
        BEGIN
                 CALL CopyMessageHeaders();
                -- CALL CopyEntireMessage();
                DECLARE cData CHAR;
                SET cData = InputRoot.XMLNSC.FileStructure.FileContent;
                SET OutputRoot.BLOB.BLOB = CAST ( cData AS BLOB ccsid 1208 );

                RETURN TRUE;
        END;

        CREATE PROCEDURE CopyMessageHeaders() BEGIN
                DECLARE I INTEGER 1;
                DECLARE J INTEGER;
                SET J = CARDINALITY(InputRoot.*[]);
                WHILE I < J DO
                        SET OutputRoot.*[I] = InputRoot.*[I];
                        SET I = I + 1;
                END WHILE;
        END;

        CREATE PROCEDURE CopyEntireMessage() BEGIN
                SET OutputRoot = InputRoot;
        END;
END MODULE;
```

Step 7:- Click on the FileOutputNode properties. Provide the Directory as “C:\IIB\Lab9\output” and File Name as “OUTPUT.TXT”



File Output Node Properties - READFLOW.OUT

Directory: C:\IIB\Lab9\output

File name or pattern: OUTPUT.TXT

File action

Mode for writing to file

○ Write directly to the output file (append if file exists)

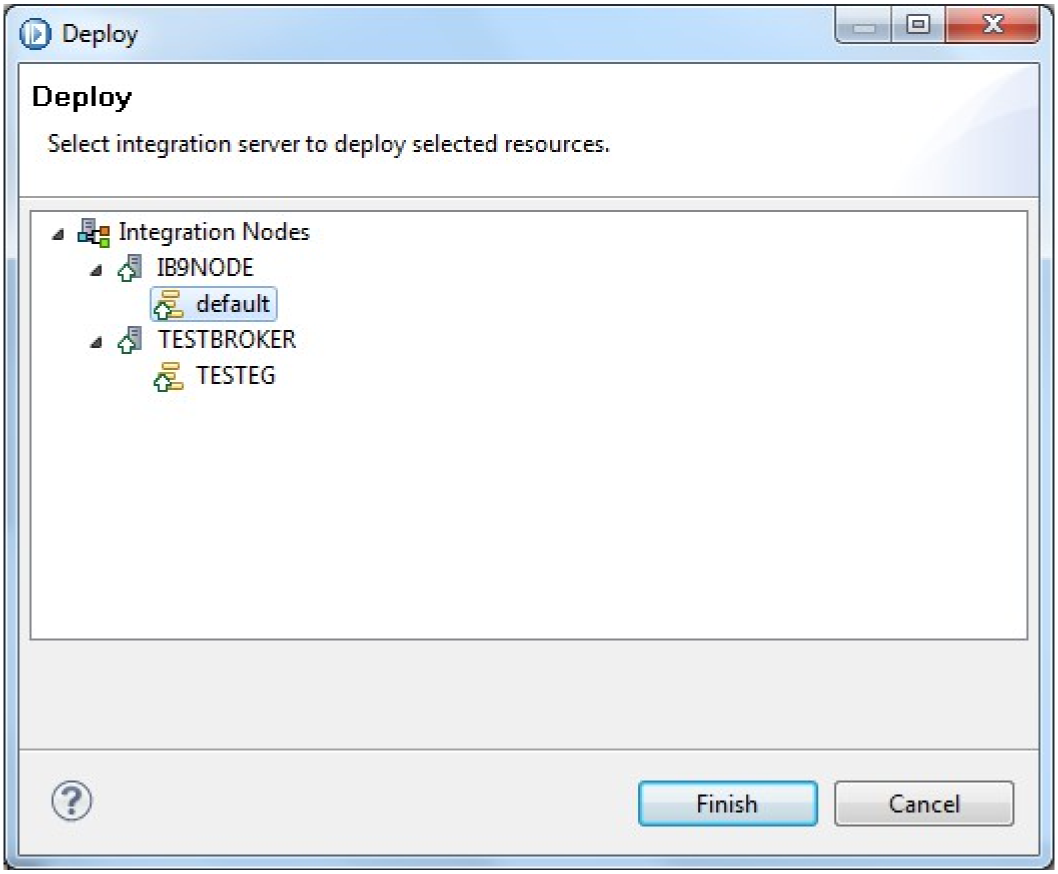
● Stage in mqsitransit directory and move to output directory on "Finish file"

Build the project.

The flow should appear like this

Step 8:- Deploy the flow.

Right click on the flow and deploy to the execution group.

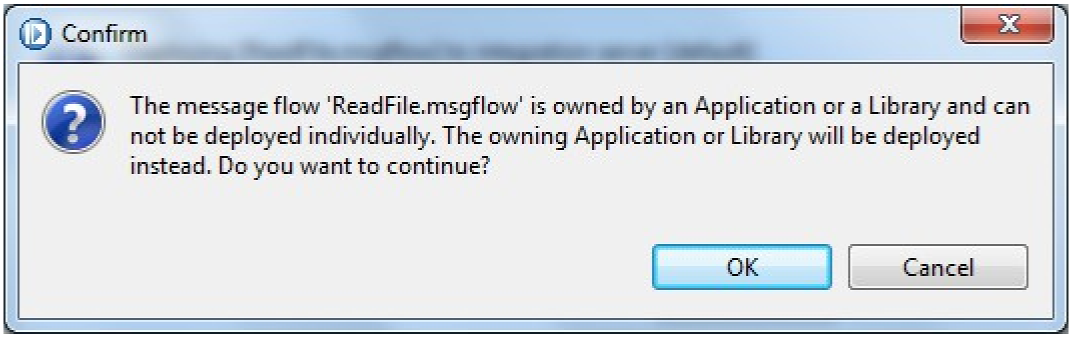


Deploy

Deploy

Select integration server to deploy selected resources.

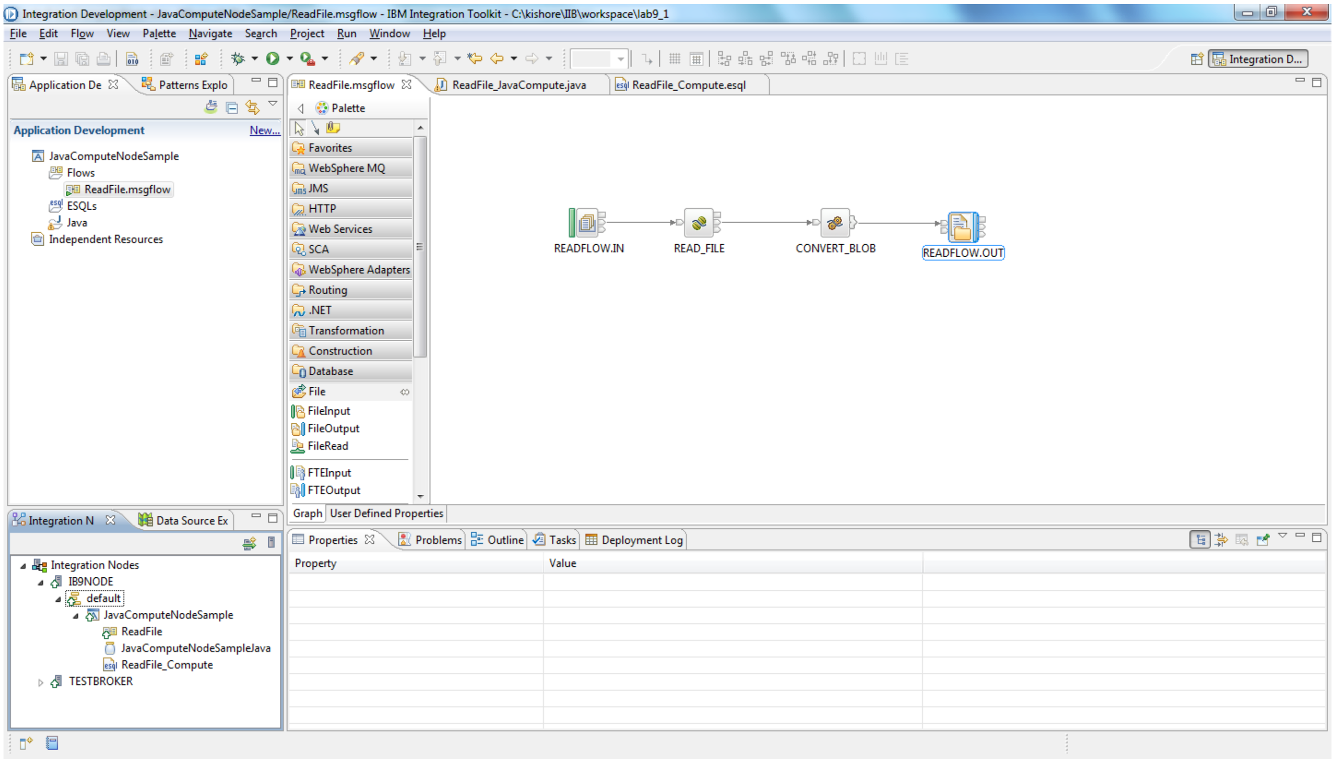
- Integration Nodes
  - IB9NODE
    - default
  - TESTBROKER
    - TESTEG



Confirm

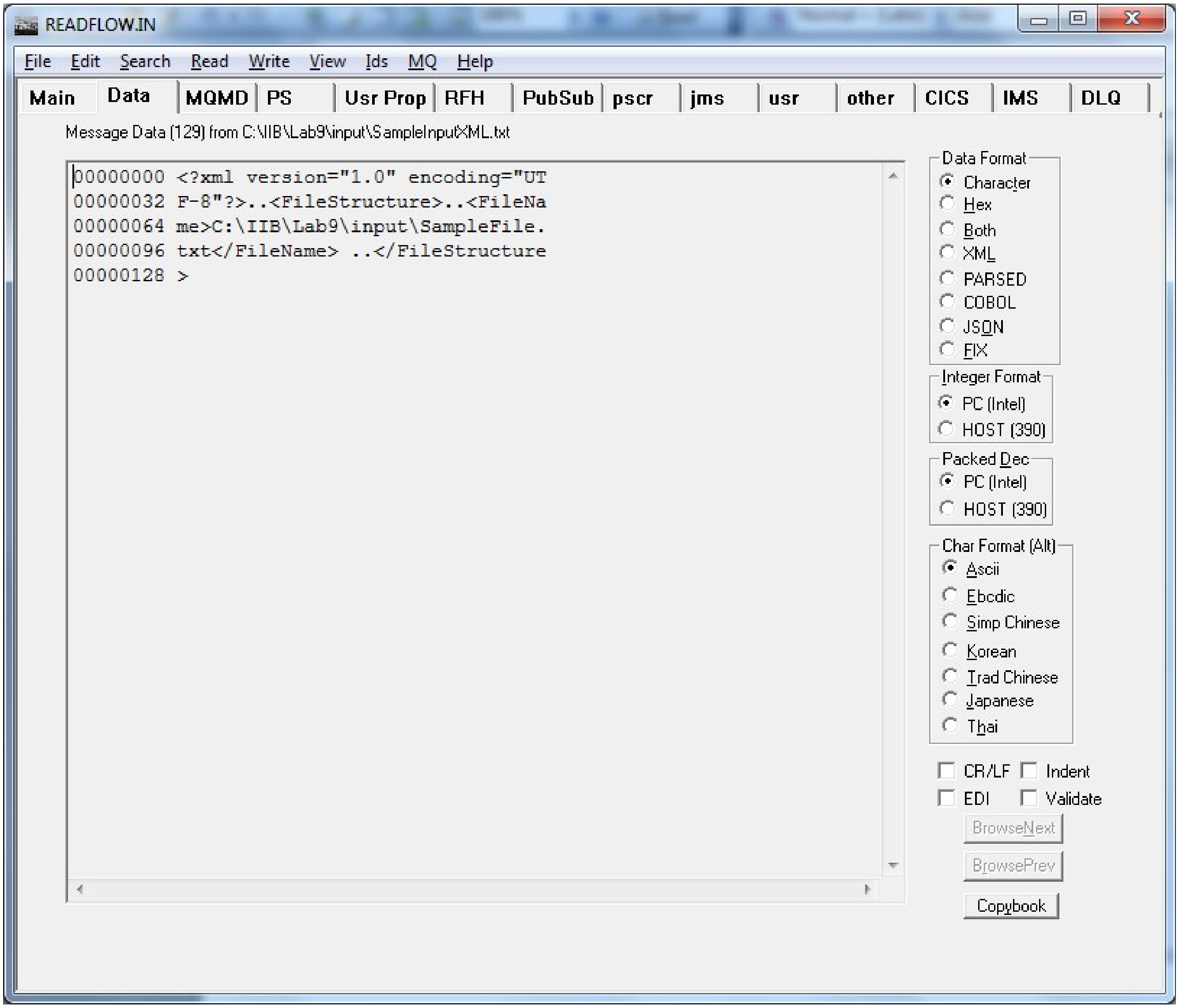
The message flow 'ReadFile.msgflow' is owned by an Application or a Library and can not be deployed individually. The owning Application or Library will be deployed instead. Do you want to continue?

Click OK


Integration Development - JavaComputeNodeSample/ReadFile.msgflow - IBM Integration Toolkit - C:\kishore\IIB\workspace\lab9_1
File Edit Flow View Palette Navigate Search Project Run Window Help
Integration D...
Application De
Patterns Explo
Application Development
New...
JavaComputeNodeSample
Flows
ReadFile.msgflow
ESQLs
Java
Independent Resources
ReadFile.msgflow
ReadFile_JavaCompute.java
ReadFile_Compute.esql
Palette
Favorites
WebSphere MQ
JMS
HTTP
Web Services
SCA
WebSphere Adapters
Routing
.NET
Transformation
Construction
Database
File
FileInput
FileOutput
FileRead
FTEInput
FTEOutput
READFLOW.IN
READ_FILE
CONVERT_BLOB
READFLOW.OUT
Graph
User Defined Properties
Integration N
Data Source Ex
Integration Nodes
IB9NODE
default
JavaComputeNodeSample
ReadFile
JavaComputeNodeSampleJava
ReadFile_Compute
TESTBROKER
Properties
Problems
Outline
Tasks
Deployment Log
Property
Value

Step 9:- Testing.
Create the required Queue "READFLOW.IN" in MQ
Copy the test resources to the lab folders C:\IIB\Lab9\input

Using RFHUtil send the input message to READFLOW.IN Queue


READFLOW.IN
File Edit Search Read Write View Ids MQ Help
Main Data MQMD PS Usr Prop RFH PubSub pscr jms usr other CICS IMS DLQ
Data Format
Integer Format
Packed Dec
Char Format (Alt)
BrowseNext
BrowsePrev
Copybook

Message Data (129) from C:\IIB\Lab9\input\SampleInputXML.txt

```
00000000 <?xml version="1.0" encoding="UT
00000032 F-8"?>..<FileStructure>..<FileNa
00000064 me>C:\IIB\Lab9\input\SampleFile.
00000096 txt</FileName> ..</FileStructure
00000128 >
```

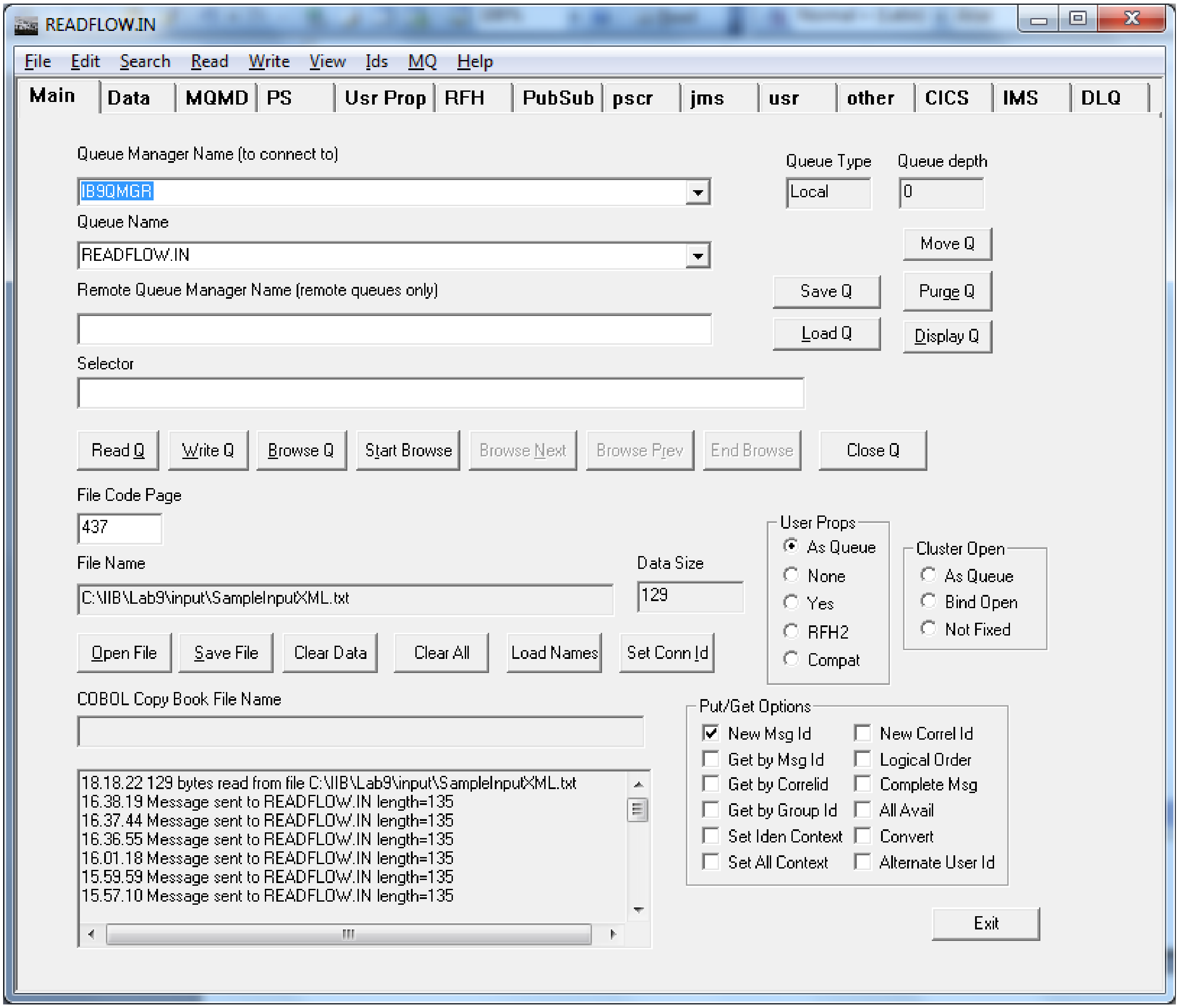
Data Format: ● Character ○ Hex ○ Both ○ XML ○ PARSED ○ COBOL ○ JSON ○ FIX

Integer Format: ● PC (Intel) ○ HOST (390)

Packed Dec: ● PC (Intel) ○ HOST (390)

Char Format (Alt): ● Ascii ○ Ebcdic ○ Simp Chinese ○ Korean ○ Trad Chinese ○ Japanese ○ Thai

□ CR/LF □ Indent
□ EDI □ Validate



Queue Manager Name (to connect to): IB9QMGR

Queue Type: Local

Queue depth: 0

Queue Name: READFLOW.IN

Remote Queue Manager Name (remote queues only):

Selector:

File Code Page: 437

File Name: C:\IIB\Lab9\input\SampleInputXML.txt

Data Size: 129

User Props
- ● As Queue
- ○ None
- ○ Yes
- ○ RFH2
- ○ Compat

Cluster Open
- ○ As Queue
- ○ Bind Open
- ○ Not Fixed

COBOL Copy Book File Name:

```
18.18.22 129 bytes read from file C:\IIB\Lab9\input\SampleInputXML.txt
16.38.19 Message sent to READFLOW.IN length=135
16.37.44 Message sent to READFLOW.IN length=135
16.36.55 Message sent to READFLOW.IN length=135
16.01.18 Message sent to READFLOW.IN length=135
15.59.59 Message sent to READFLOW.IN length=135
15.57.10 Message sent to READFLOW.IN length=135
```

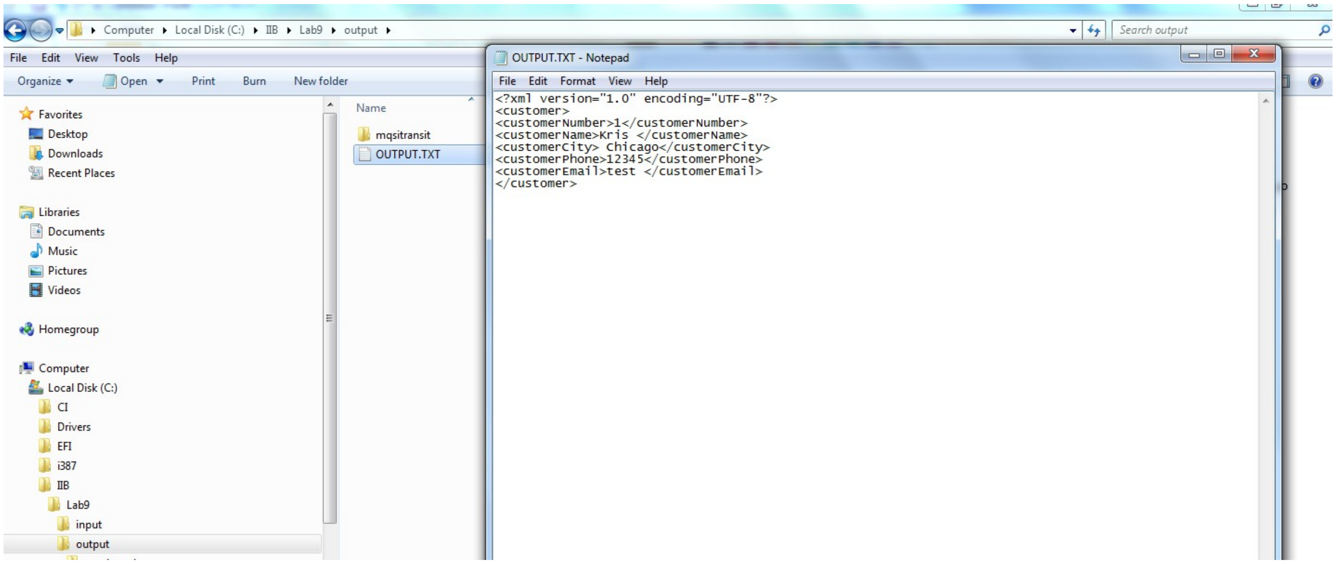
Put/Get Options

☑ New Msg Id	□ New Correl Id
□ Get by Msg Id	□ Logical Order
□ Get by Correlid	□ Complete Msg
□ Get by Group Id	□ All Avail
□ Set Iden Context	□ Convert
□ Set All Context	□ Alternate User Id

Click on Write Q to send the message.

Result:

Output file is generate in the specified directory C:\IIB\Lab9\output



```
<?xml version="1.0" encoding="UTF-8"?>
<customer>
<customerNumber>1</customerNumber>
<customerName>Kris </customerName>
<customerCity> Chicago</customerCity>
<customerPhone>12345</customerPhone>
<customerEmail>test </customerEmail>
</customer>
```