Configuring Eclipse to develop Java/JMS programs for MQ 8.0

IBM Techdoc: 7046013

http://www.ibm.com/support/docview.wss?uid=swg27046013

Date last updated: 30-Jun-2015

Christian Montes - comontes@us.ibm.com
Angel Rivera – rivera@us.ibm.com
IBM MQ Support

++++ Objective

To provide a tutorial on how to use the Eclipse platform to import and run MQ samples programs that exploit the MQ classes for Java and MQ classes for JMS for IBM MQ 8.0.

This tutorial assumes that MQ 8.0 is already installed in the computer.

The configuration used for the development of this techdoc is:
Windows 7, 64-bit, using MQ 8.0.0 Fix Pack 2 (8.0.0.2)

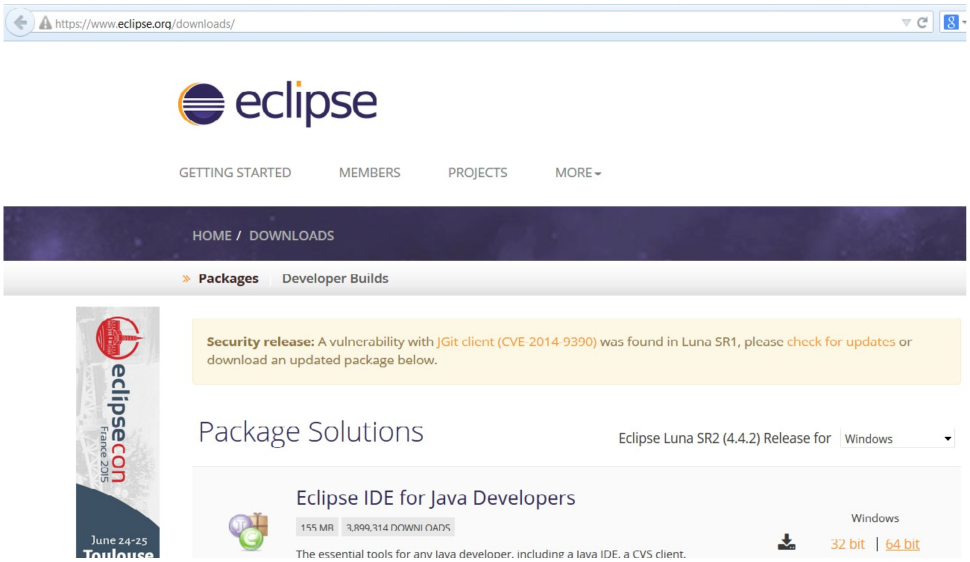
The chapters are:

Chapter 1: Downloading and Installing Eclipse in Windows

Login to your Windows machine and identify an existing folder where you are going to download a zip file that will contain the Eclipse code, such as c:\downloads or c:\eclipse-mq
Identify another directory where you are going to extract the files from the zip file. The extracted files will be extracted under a subdirectory called "eclipse\". You could use the same directory for the downloaded zip file and the extracted files.

Download Eclipse IDE for Java Developers for your platform from:

https://www.eclipse.org/downloads/



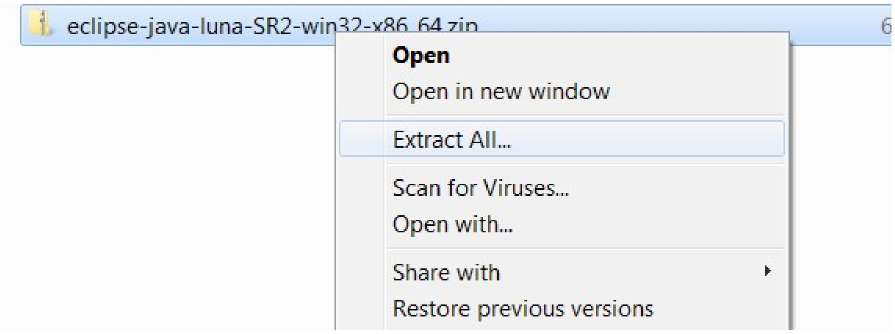
Security release: A vulnerability with JGit client (CVE 2014 9390) was found in Luna SR1, please check for updates or download an updated package below.

The essential tools for any Java developer, including a Java IDE, a CVS client,

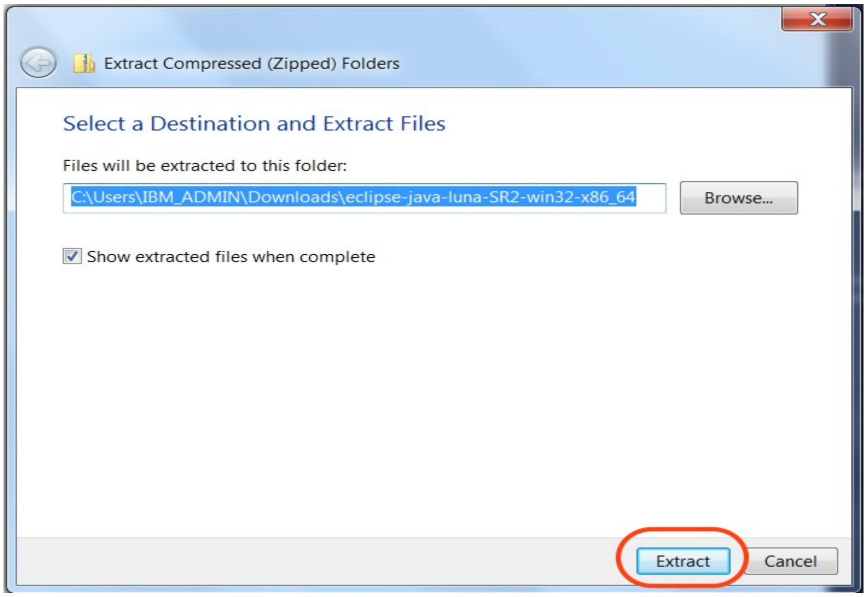
For this document the following file was downloaded:

eclipse-java-luna-SR2-win32-x86_64.zip

Unzip the downloaded zip file.
One method is to right click on the downloaded zip file and click:
Extract all.



Press ***Extract*** and wait to finish.



Extract Compressed (Zipped) Folders

Select a Destination and Extract Files

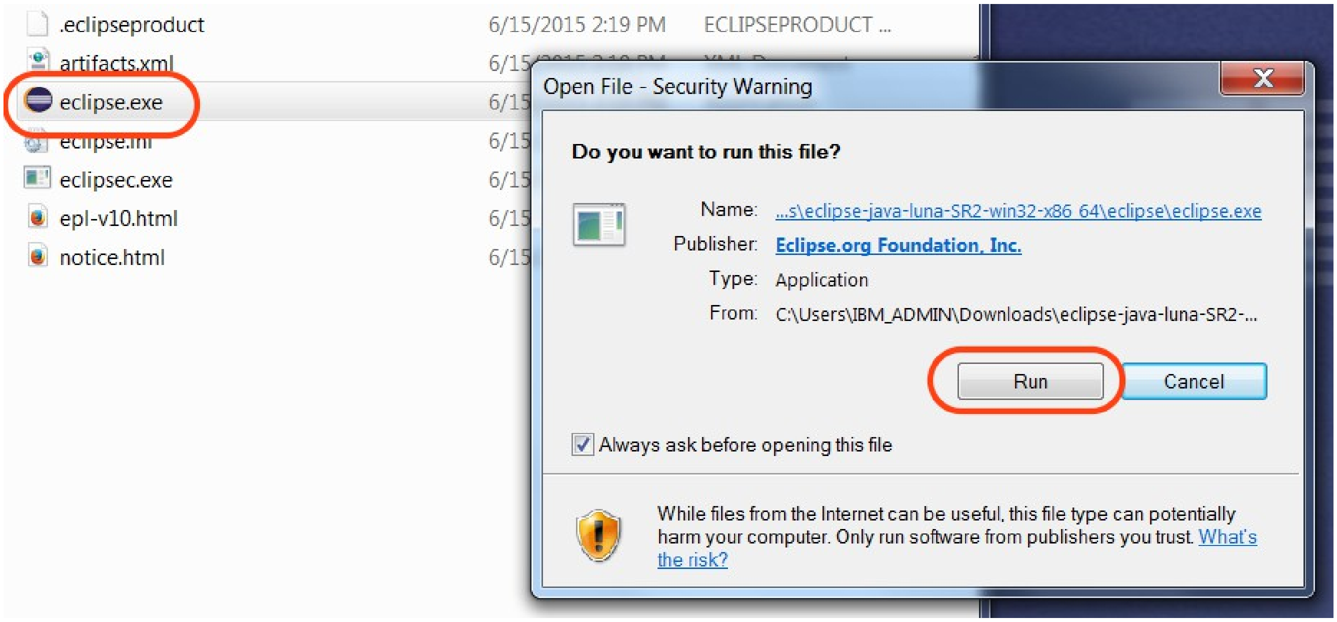
Files will be extracted to this folder:

C:\Users\IBM_ADMIN\Downloads\eclipse-java-luna-SR2-win32-x86_64

☑ Show extracted files when complete

Open the new created folder ***newFolderName/eclipse/***

Double click on the file ***eclipse.exe*** and then press ***Run***.



Open File - Security Warning

Do you want to run this file?

Name: ...s\eclipse-java-luna-SR2-win32-x86_64\eclipse\eclipse.exe

Publisher: **Eclipse.org Foundation, Inc.**

Type: Application

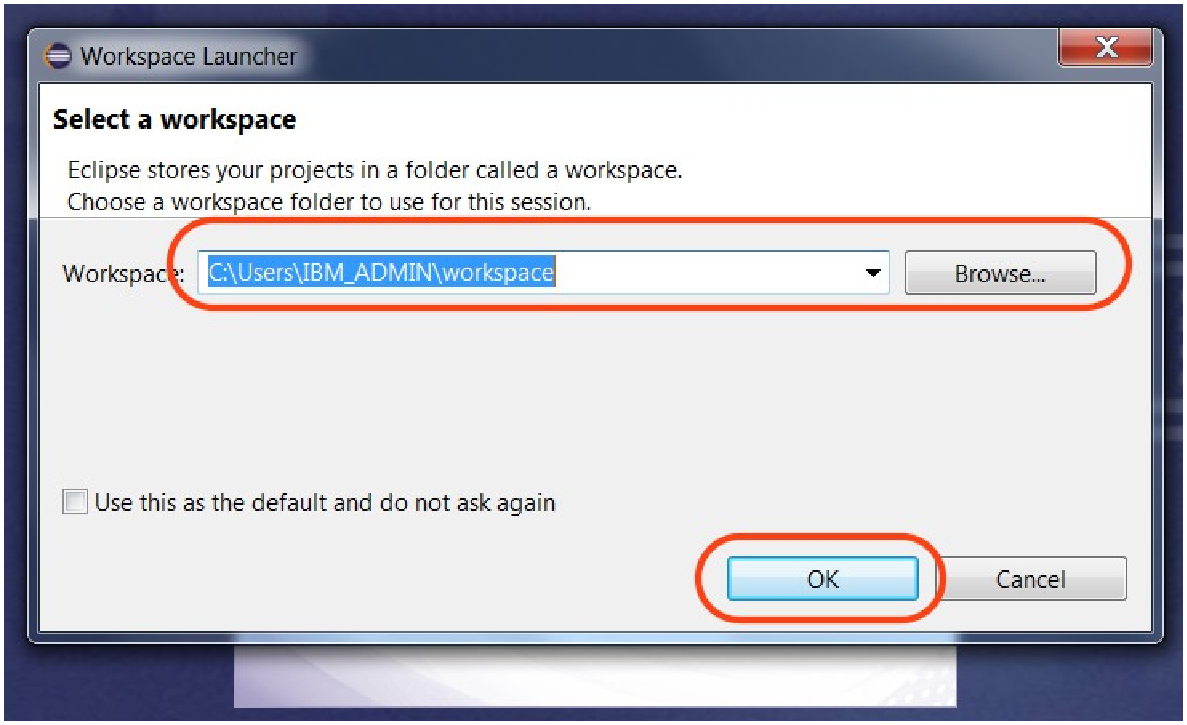
From: C:\Users\IBM_ADMIN\Downloads\eclipse-java-luna-SR2-...

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

The *Eclipse* application is now starting.

Choose the default workspace by clicking the ***OK*** button or select other workspace folder by clicking on ***Browse*** and then ***OK***.



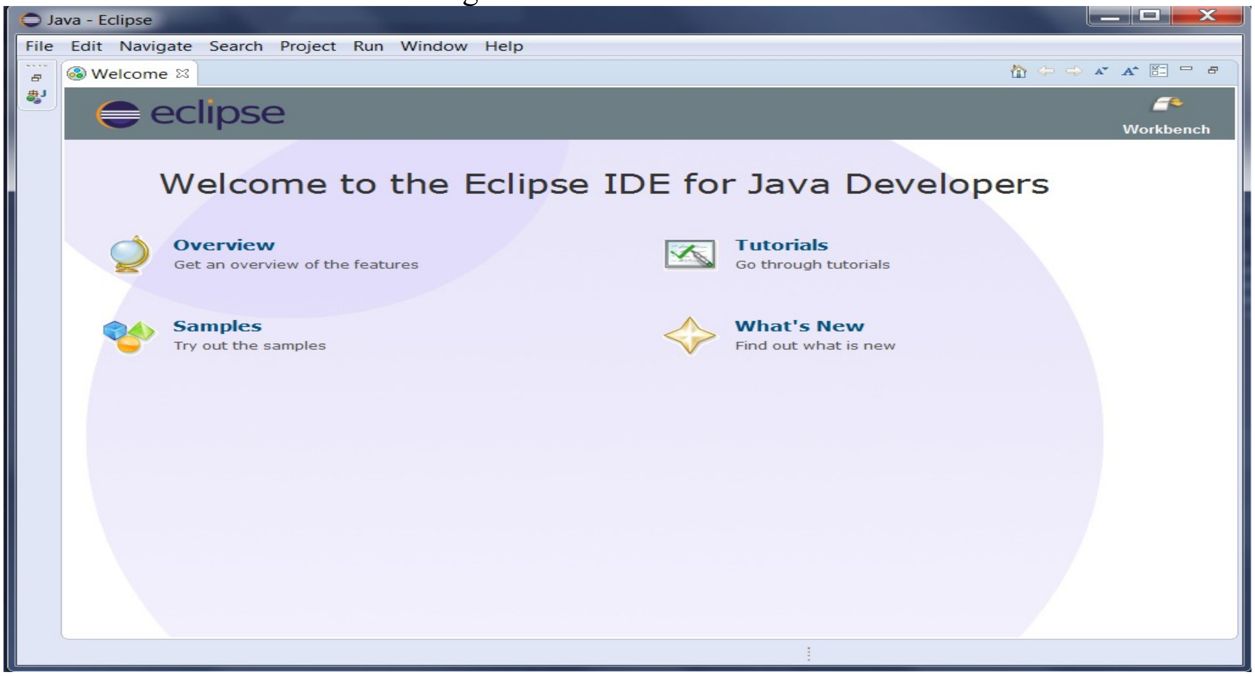
Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\Users\IBM_ADMIN\workspace

□ Use this as the default and do not ask again

You will be welcome with the following screen.



Welcome to the Eclipse IDE for Java Developers

At this point, the installation is complete. You can now proceed to the next chapter to import a sample program.

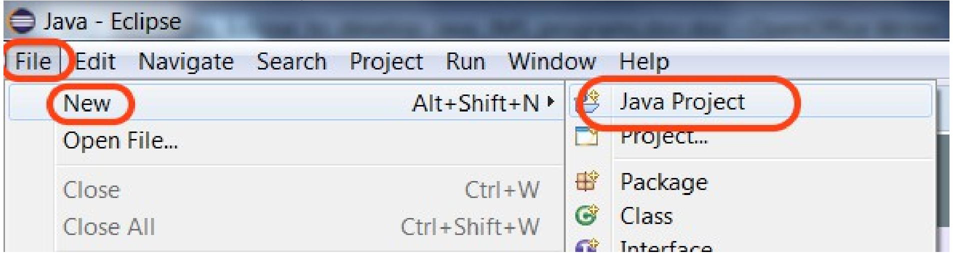
Chapter 2: Importing and running a sample MQ Java program

Section 2.1: Importing an MQ Java sample.

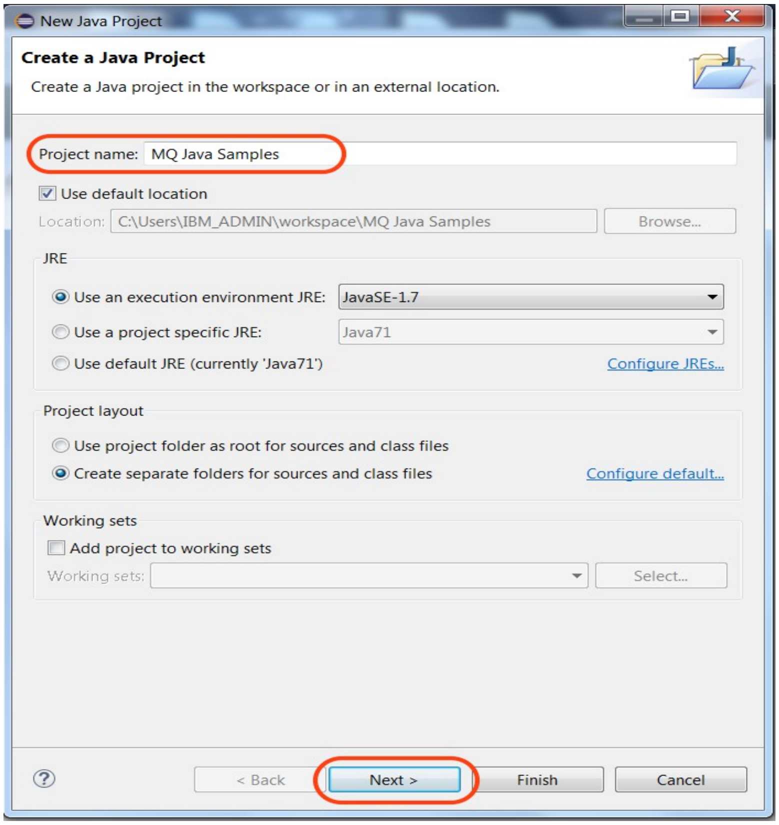
MQ ships some sample programs and this chapter describes how you can import into Eclipse an MQ Java program that is located in the directory:

C:\Program Files\IBM\WebSphere MQ\Tools\wmqjava\samples

1. Select ***File → New → Java Project***



2. In the ***Project name*** field type: ***MQ Java Samples.*** Accept the defaults and click ***Next***.


New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: MQ Java Samples

☑ Use default location

Location: C:\Users\IBM_ADMIN\workspace\MQ Java Samples

JRE

● Use an execution environment JRE: JavaSE-1.7

○ Use a project specific JRE: Java71

○ Use default JRE (currently 'Java71')

Project layout

○ Use project folder as root for sources and class files

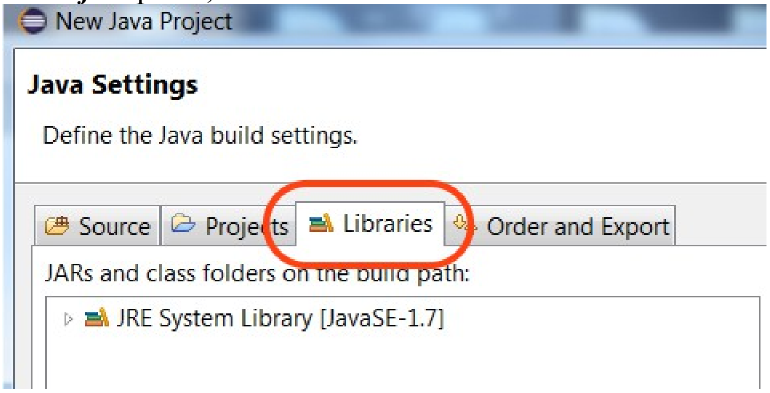
● Create separate folders for sources and class files

Working sets

□ Add project to working sets

Working sets:

3. In the ***New Java Project*** panel, switch to the ***Libraries*** tab.



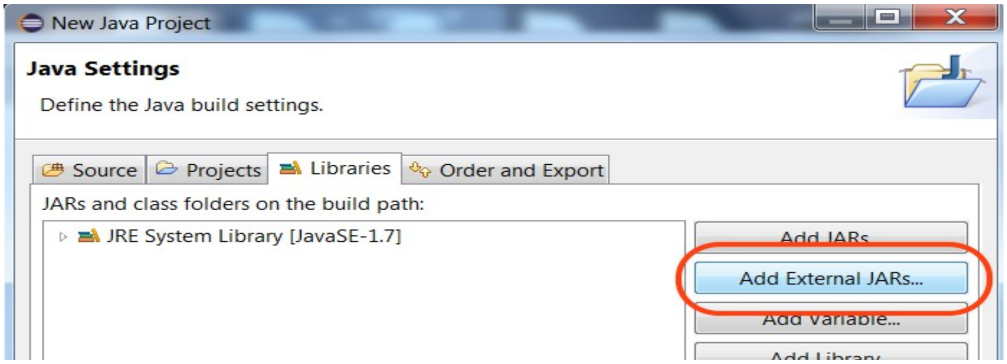
Java Settings

Define the Java build settings.

JARs and class folders on the build path:

JRE System Library [JavaSE-1.7]

4. Click on ***Add External JARs…***



Java Settings

Define the Java build settings.

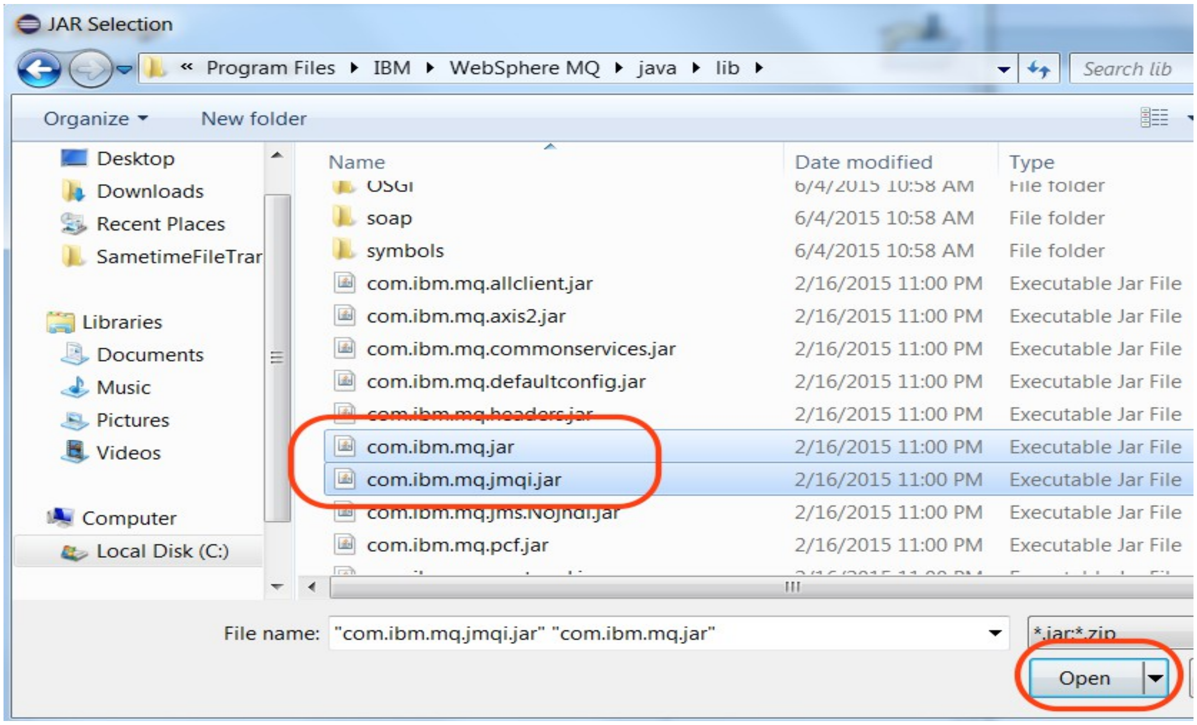
JARs and class folders on the build path:

JRE System Library [JavaSE-1.7]

5. Browse to:
   C:\Program Files\IBM\Websphere MQ\java\lib

   ... and select for Java (not for JMS):
   com.ibm.mq.jar
   com.ibm.mq.jmqi.jar

   Note for JMS: you would need to select:
   com.ibm.mq.jms.Nojndi.jar
   com.ibm.mqjms.jar

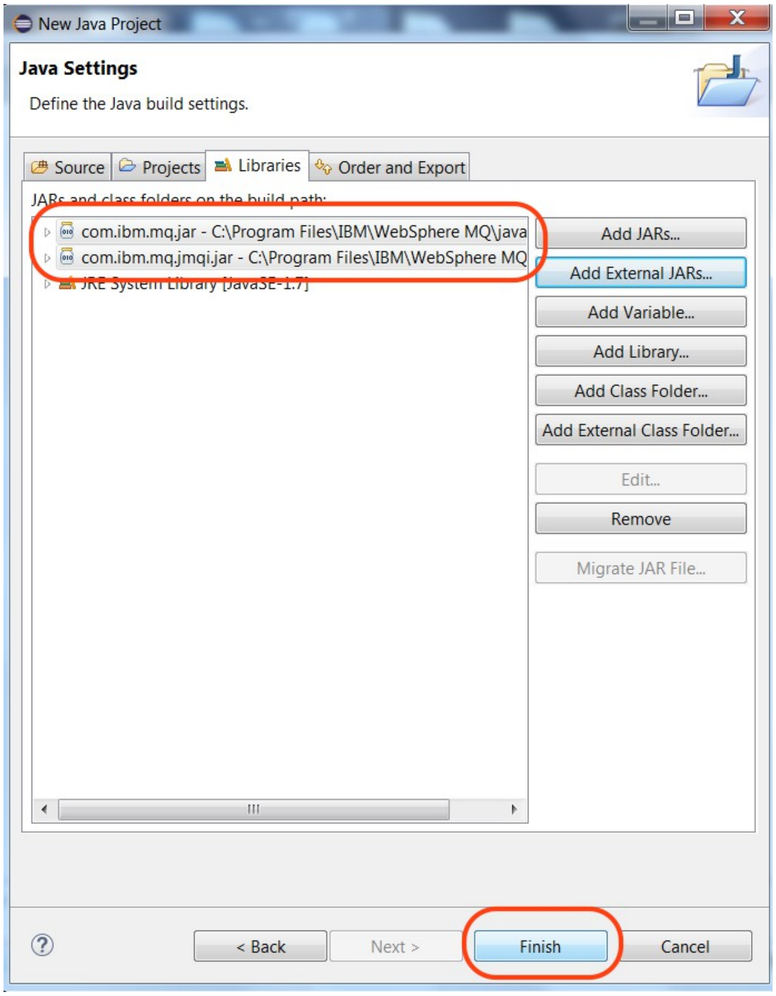


Name	Date modified	Type
OSGi	6/4/2015 10:58 AM	File folder
soap	6/4/2015 10:58 AM	File folder
symbols	6/4/2015 10:58 AM	File folder
com.ibm.mq.allclient.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.axis2.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.commonservices.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.defaultconfig.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.headers.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.jmqi.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.jms.Nojndi.jar	2/16/2015 11:00 PM	Executable Jar File
com.ibm.mq.pcf.jar	2/16/2015 11:00 PM	Executable Jar File

File name: "com.ibm.mq.jmqi.jar" "com.ibm.mq.jar"

.jar;.zip

6. Click ***Open***, the libraries will be added.



New Java Project

Java Settings

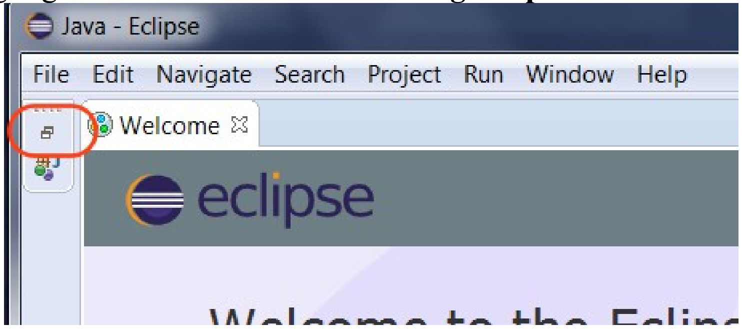
Define the Java build settings.

JARs and class folders on the build path:

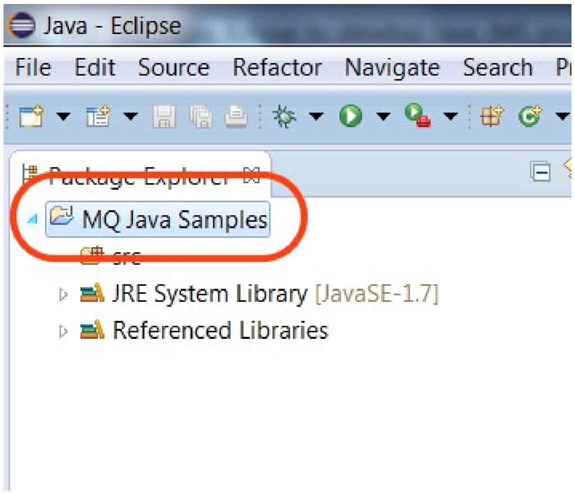
- com.ibm.mq.jar - C:\Program Files\IBM\WebSphere MQ\java
- com.ibm.mq.jmqi.jar - C:\Program Files\IBM\WebSphere MQ
- JRE System Library [JavaSE-1.7]

Click ***Finish***.

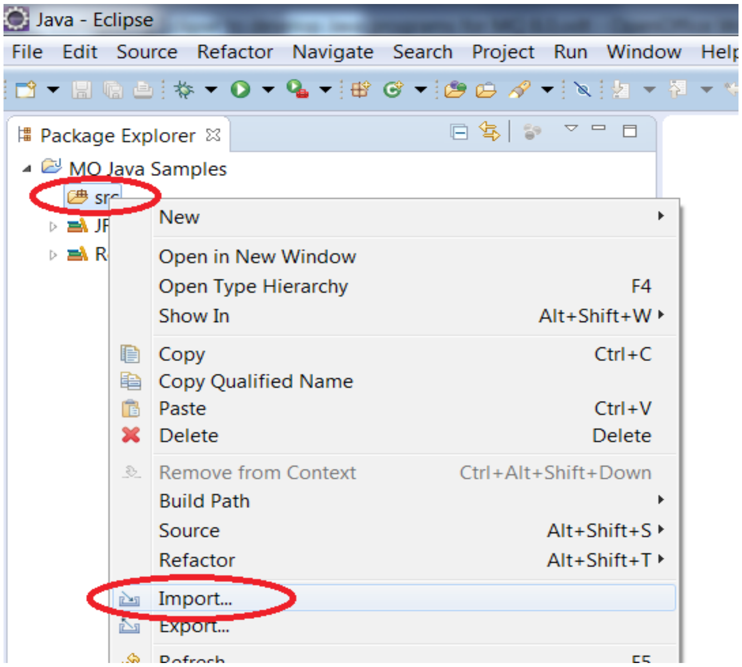
7. Click on the highlighted button to show the **Package Explorer** view.



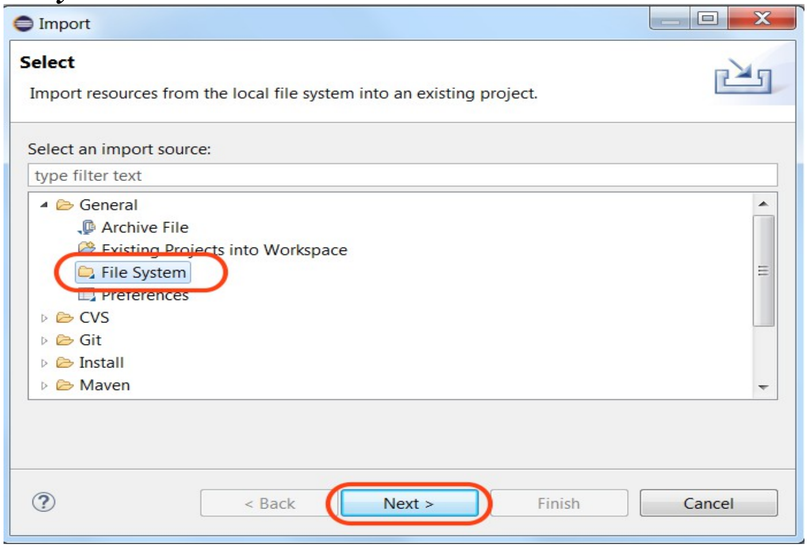
8. Notice that the new project "***MQ Java Samples***" is shown in the Package Explorer view.


Java - Eclipse
File Edit Source Refactor Navigate Search Pr
Package Explorer
MQ Java Samples
src
JRE System Library [JavaSE-1.7]
Referenced Libraries

9. Right-click in the ***src*** folder from the "***MQ Java Samples***" project in the ***Package Explorer*** view.
Select ***Import…***



10. In the Import window select:
    General → File System



Import

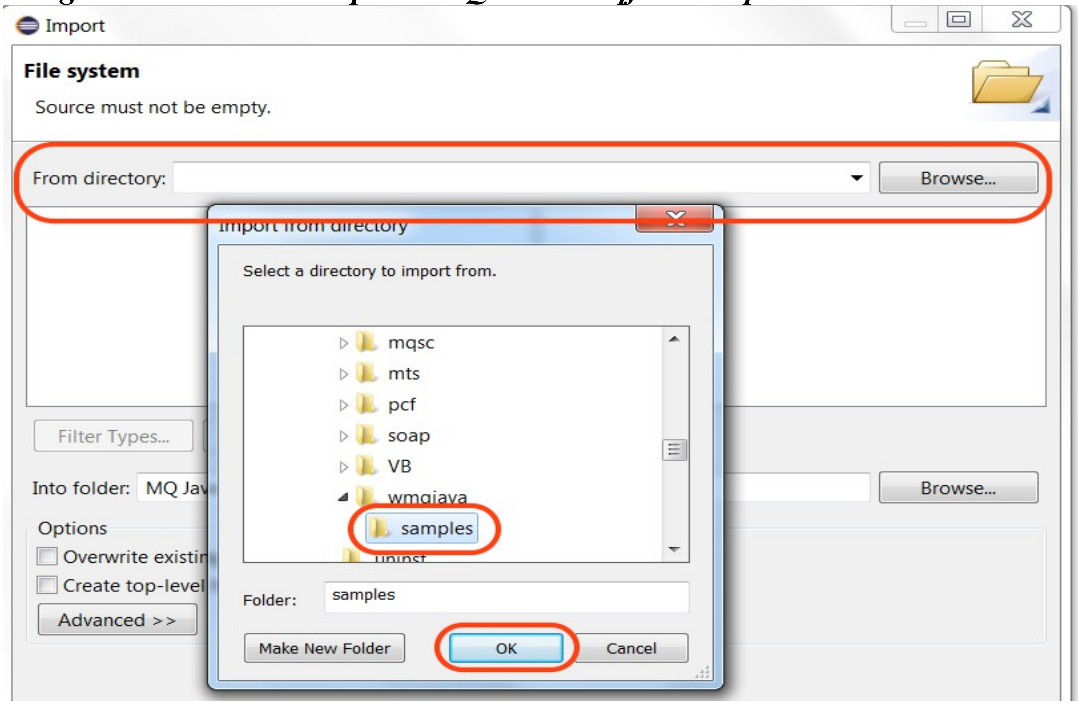
Select

Import resources from the local file system into an existing project.

Select an import source:

Click ***Next***.

11. In the "***File system***" panel, Click ***Browse...*** and browse to the path:
    C:\Program Files\IBM\Websphere MQ\tools\wmqjava\samples



Import

File system

Source must not be empty.

From directory:

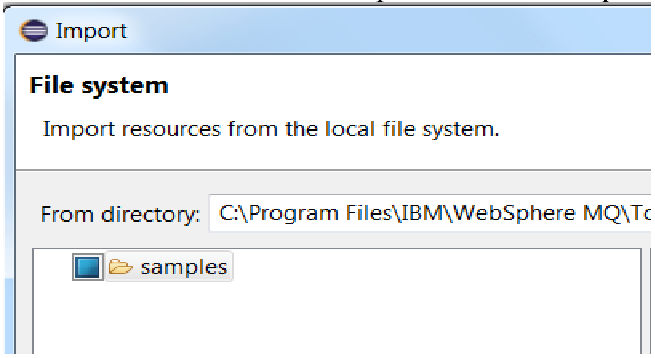
Import from directory

Select a directory to import from.

Folder: samples

Click ***OK***

12. In the left pane of the Import panel, click on "***samples***"
    Do not select the check box because that will import ALL the samples.



Import

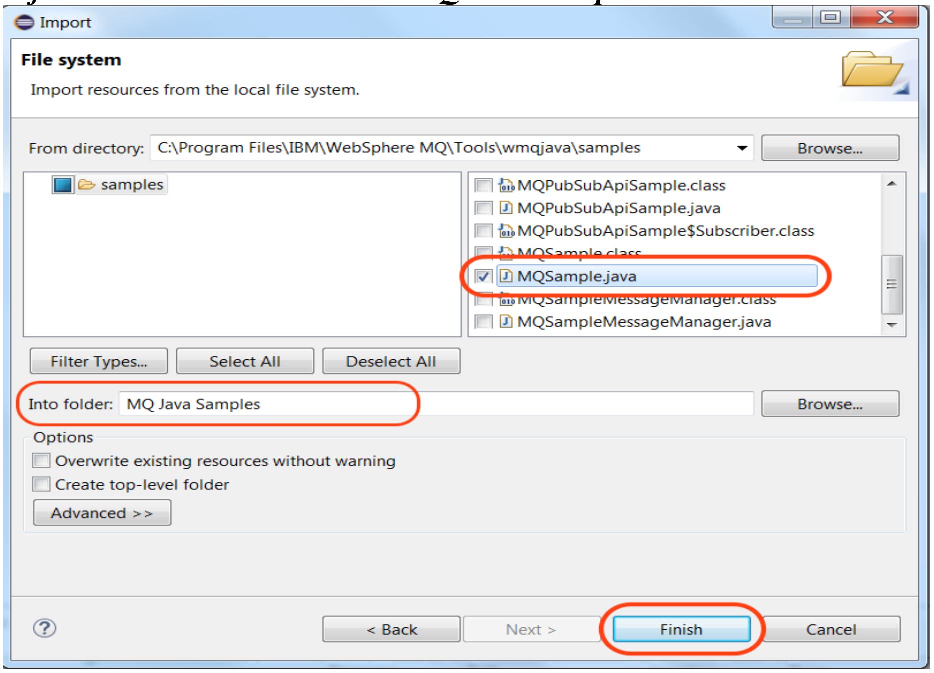
File system

Import resources from the local file system.

From directory: C:\Program Files\IBM\WebSphere MQ\Tc

□ samples

13. In the right pane, select: ***MQSample.java*** .
    The "***Into folder***" field should contain "***MQ Java Samples/src***".



Import

File system

Import resources from the local file system.

From directory: C:\Program Files\IBM\WebSphere MQ\Tools\wmqjava\samples

□ samples

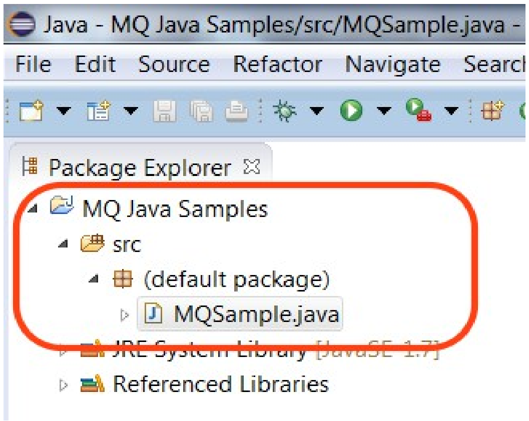
□ MQPubSubApiSample.class
□ MQPubSubApiSample.java
□ MQPubSubApiSample$Subscriber.class
□ MQSample.class
☑ MQSample.java
□ MQSampleMessageManager.class
□ MQSampleMessageManager.java

Into folder: MQ Java Samples

Options
□ Overwrite existing resources without warning
□ Create top-level folder

Click ***Finish***.

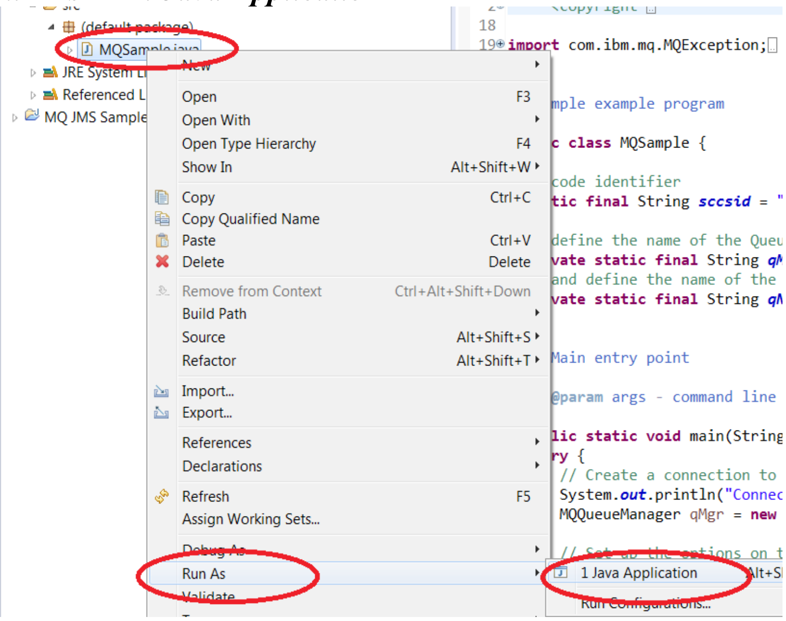
14. From the ***Package Explorer***, expand "***MQ Java Samples***".
    Notice "(default package)"
    Expand it to see the Java source code.


Java - MQ Java Samples/src/MQSample.java -
File Edit Source Refactor Navigate Searc
Package Explorer
MQ Java Samples
src
(default package)
MQSample.java
JRE System Library [JavaSE-1.7]
Referenced Libraries

Section 2.1.1: How to run the sample program.

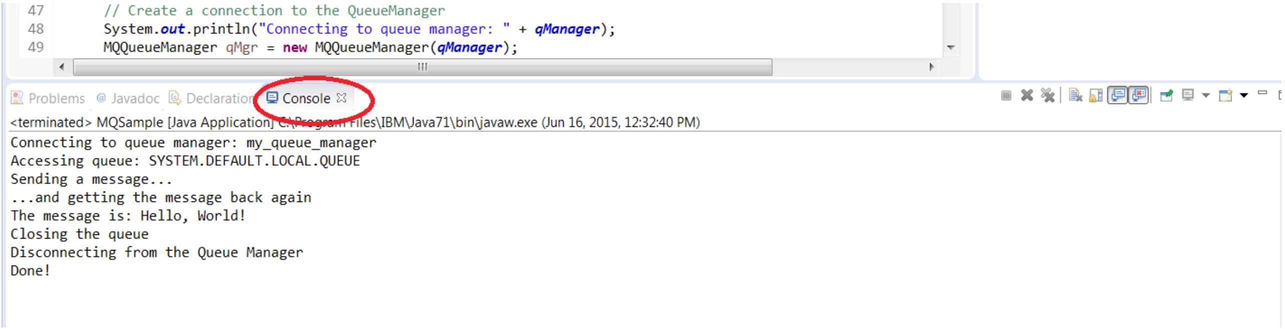
For this example, the file does not require any parameters. You can follow the following steps to run it.

1. In the default package under the project "***MQ Java Samples***", right-click on ***MQSample.java***.
   Then select: ***Run As → 1. Java Application***



```
import com.ibm.mq.MQException;
mple example program
c class MQSample {
code identifier
tic final String sccsid = "
define the name of the Queu
vate static final String qM
and define the name of the
vate static final String qM
Main entry point
@param args - command line
lic static void main(String
ry {
 // Create a connection to
 System.out.println("Connec
 MQQueueManager qMgr = new
 // Set up the options on t
```

2. The results will be shown in the “Console” tab.



```
        // Create a connection to the QueueManager
        System.out.println("Connecting to queue manager: " + qManager);
        MQQueueManager qMgr = new MQQueueManager(qManager);
```

```
<terminated> MQSample [Java Application] C:\Program Files\IBM\Java71\bin\javaw.exe (Jun 16, 2015, 12:32:40 PM)
Connecting to queue manager: my_queue_manager
Accessing queue: SYSTEM.DEFAULT.LOCAL.QUEUE
Sending a message...
...and getting the message back again
The message is: Hello, World!
Closing the queue
Disconnecting from the Queue Manager
Done!
```

You can double-click on the “***Console***” tab to expand it to better see all the messages.

If there are runtime errors:

3. Reason Code: 2495

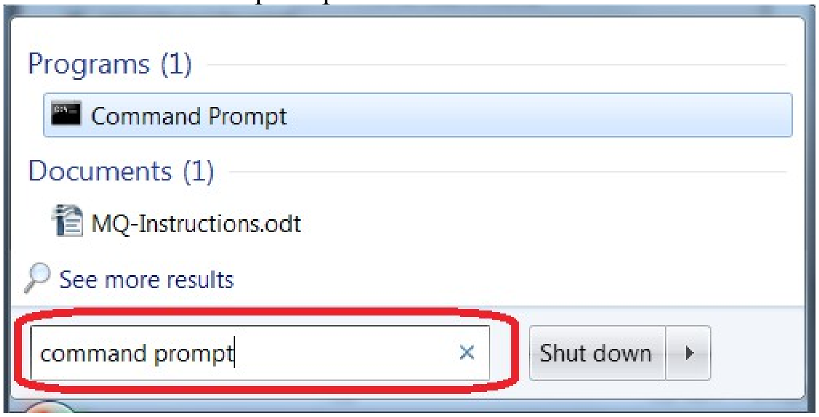
```
Connecting to queue manager: my_queue_manager
A WebSphere MQ Error occured : Completion Code 2 Reason Code 2495
com.ibm.mq.MQException: MQJE001: Completion Code '2', Reason '2495'.
```

Workaround:
Open command prompt, then go to the directory where the executable eclipse.exe is located, such as:
C:\eclipse-luna\eclipse >
Run the MQ setmqenv command, such as:
"C:\Program Files\IBM\WebSphere MQ\bin\setmqenv" -n Installation1
Then issue: eclipse.exe

Another workaround is that if you have only one version of MQ, you can make that version the Primary one, thus, avoiding the need to use setmqenv.
"C:\Program Files\IBM\WebSphere MQ\bin\setmqinst" -i -n Installation1

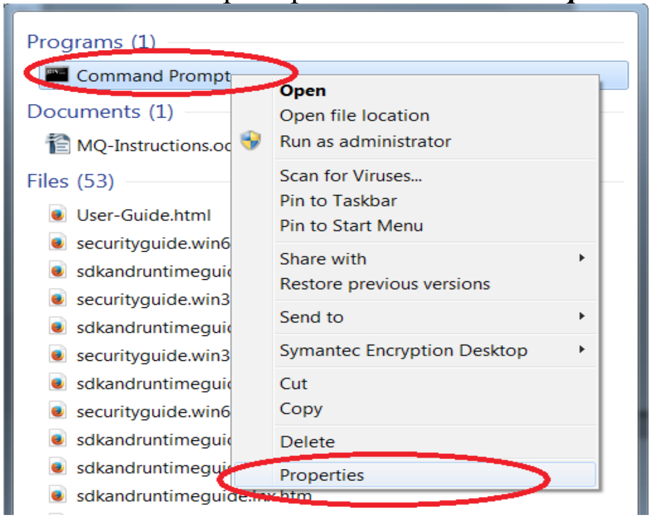
Another option is to run the ***setmqenv*** command automatically in a new command prompt:

1. Search for the command prompt on the Windows Search.

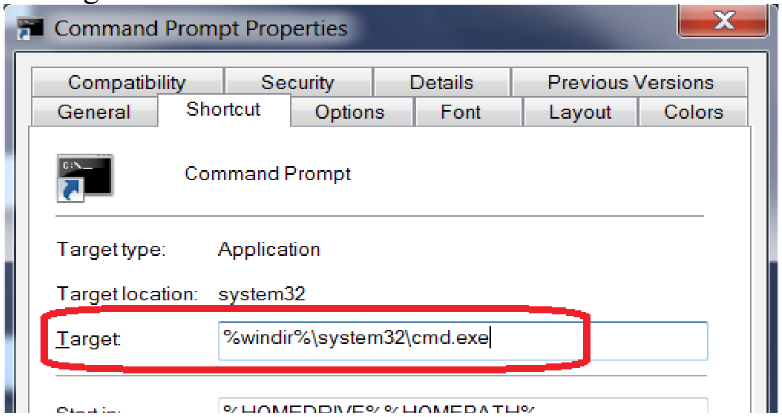


command prompt

2. Right-click on the command prompt icon and select ***Properties***.



3. Go to the Target field.



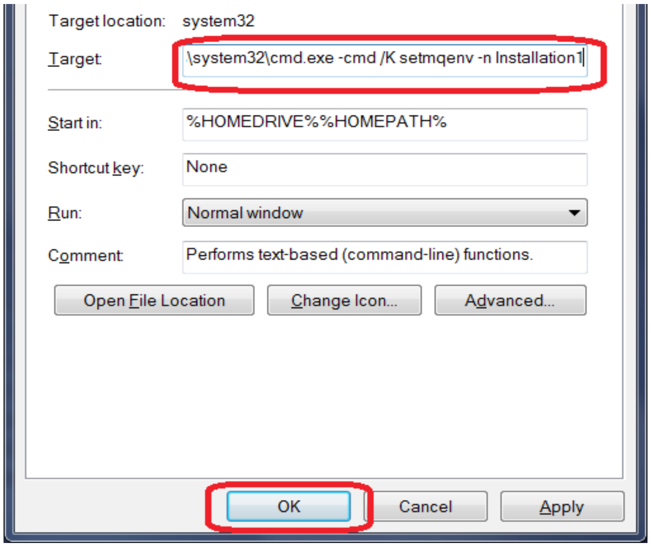
Target type: Application

Target location: system32

Target: %windir%\system32\cmd.exe

Start in: %HOMEDRIVE%%HOMEPATH%

Add the following to the end: ***-cmd /K setmqenv -n Installation1***



Target location: system32

Target: \system32\cmd.exe -cmd /K setmqenv -n Installation1

Start in: %HOMEDRIVE%%HOMEPATH%

Shortcut key: None

Run: Normal window

Comment: Performs text-based (command-line) functions.

Click **OK**.

   4. Now, every time that you open a new command prompt, the setmqenv command will be run automatically.

4. Reason Code: 2058

```
       Connecting to queue manager: my_queue_manager
       A WebSphere MQ Error occured:  Completion Code 2 Reason Code 2058
       com.ibm.mq.MQException: MQJE001: Completion Code '2', Reason '2058'.
```

   The reason code “2058” has a short name that can be found as follows:
   C:> ***mqrc 2058***
   2058 0x0000080a MQRC_Q_MGR_NAME_ERROR

   This means that the queue manager does not exist or it exists but it could not be located.
   See Section 2.1.2 which shows how to create a queue manager.|

5. If you have already a queue manager in your PC, then you can modify the following line from the MQSample.java source.
   For example, if the name of the existing queue manager is "QM_80", replace "my_queue_manager" with "QM_80":

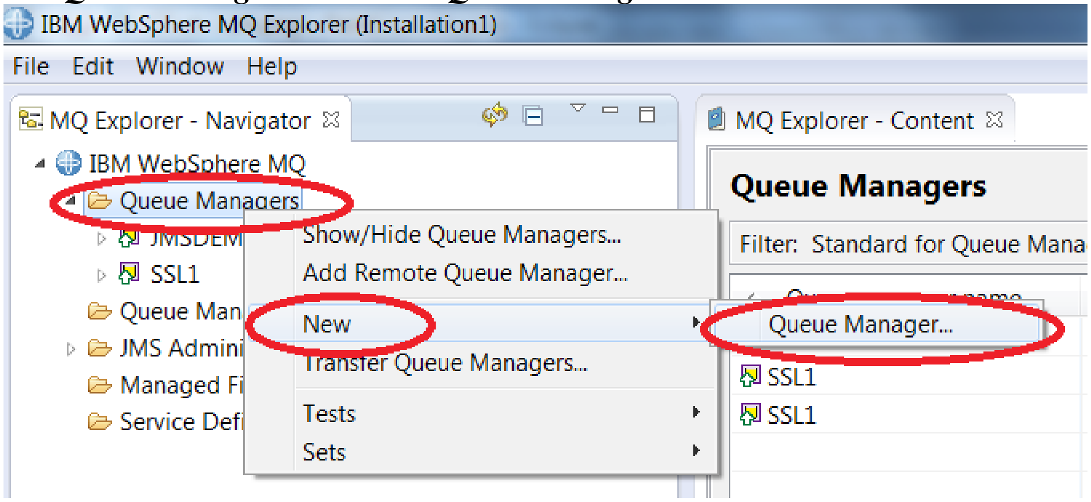
```
 // define the name of the QueueManager
  private static final String qManager = "QM_80";
```

6. Save the changes to the file MQSample.java and then run it again as an application.
   This time the messages should look like this:

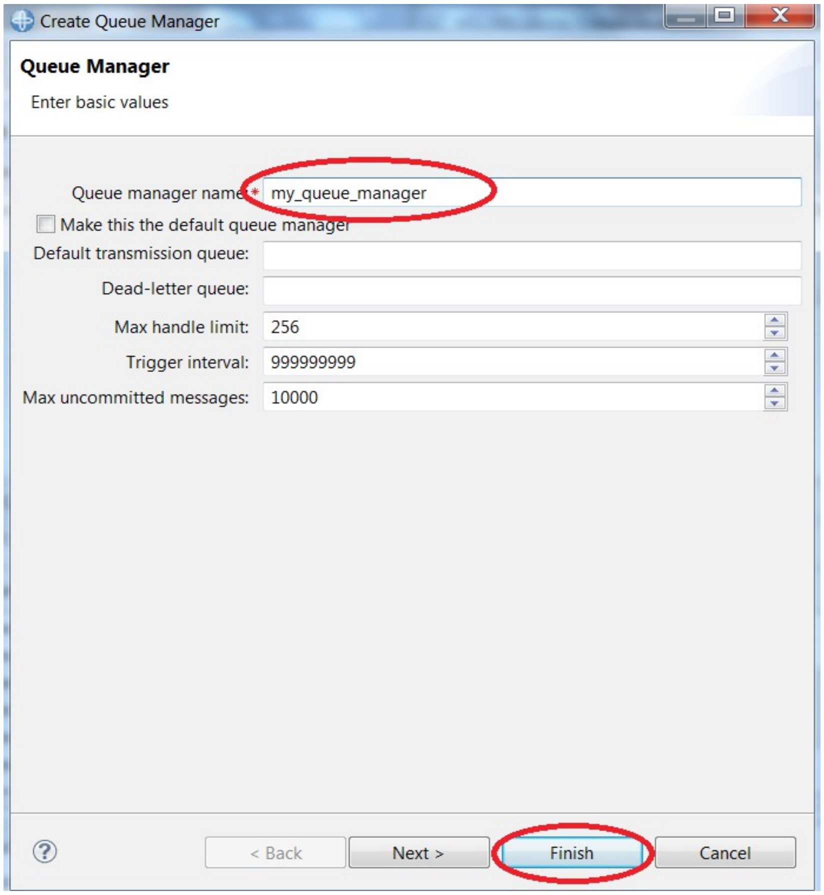
```
Connecting to queue manager: QM_80
Accessing queue: SYSTEM.DEFAULT.LOCAL.QUEUE
Sending a message...
...and getting the message back again
The message is: Hello, World!
Closing the queue
Disconnecting from the Queue Manager
Done!
```

Section 2.1.2: How to create a queue manager.

1. To create the Queue Manager, go to the MQ Explorer view:.
   Select ***Queue Managers*** → ***New*** → ***Queue Manager...***



2. In the ***Create Queue Manager*** panel, enter the name: ***my_queue_manager*** .
   Accept the defaults and click ***Finish***.



Queue manager name: * my_queue_manager

□ Make this the default queue manager

Default transmission queue:

Dead-letter queue:

Max handle limit: 256

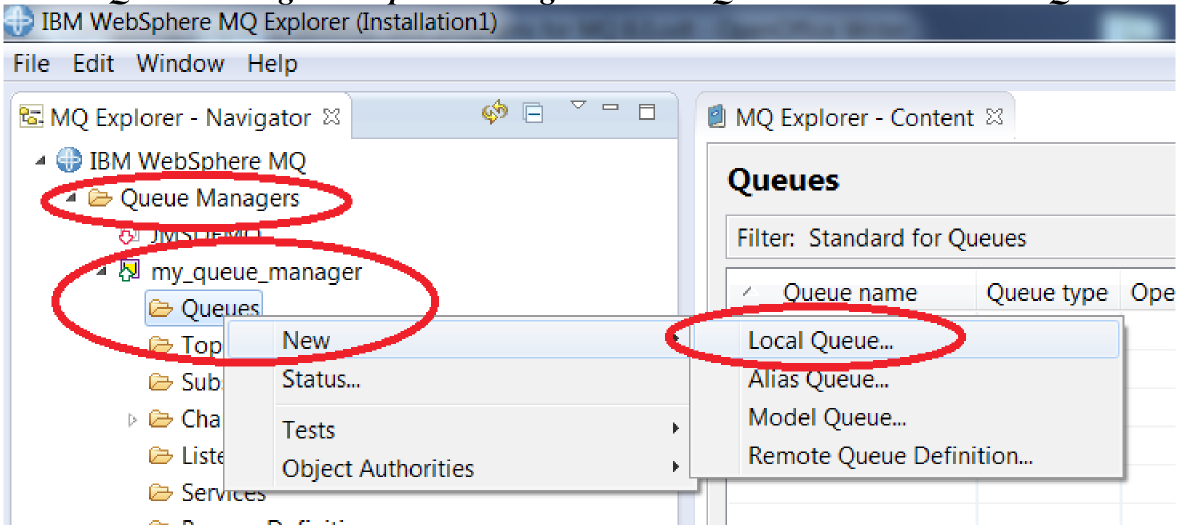
Trigger interval: 999999999

Max uncommitted messages: 10000

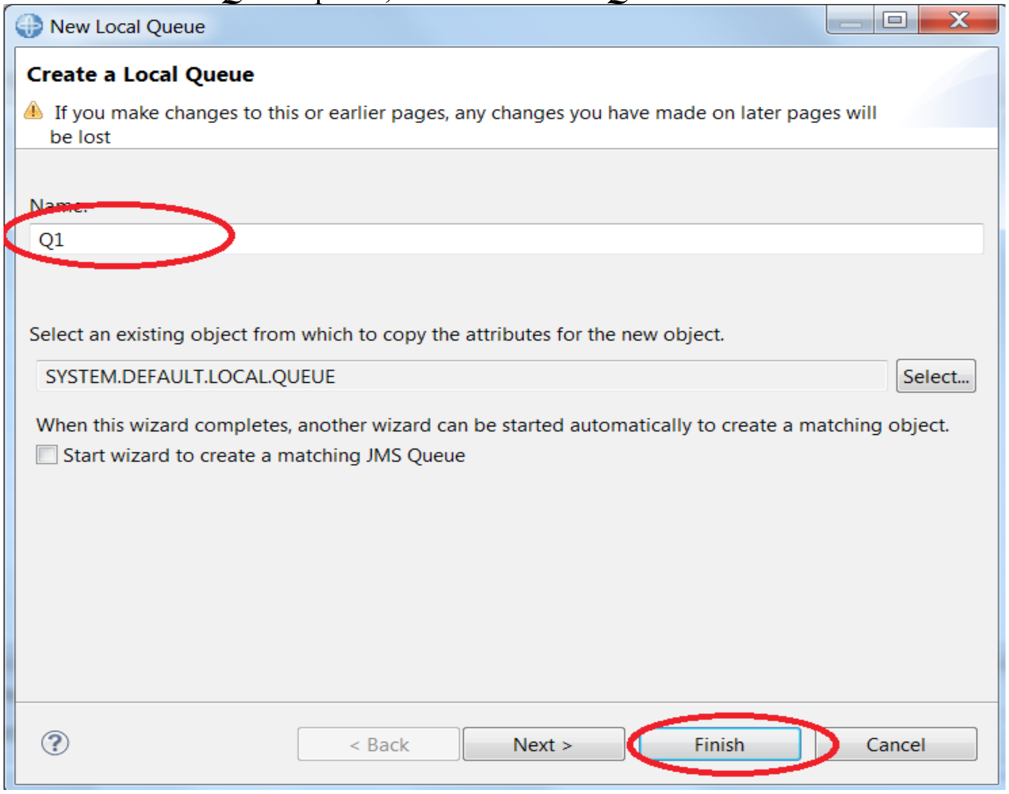
3. Run again the Java application as described before.

Section 2.1.3: How to create a local queue.

1. To create the Local Queue, go to the MQ Explorer view:.
   Select ***Queue Managers → queueManagerName → Queues → New → Local Queue...***



2. In the ***Create a Local Queue*** panel, enter the name: ***Q1*** .



Create a Local Queue

⚠ If you make changes to this or earlier pages, any changes you have made on later pages will be lost

Name:

Q1

Select an existing object from which to copy the attributes for the new object.

SYSTEM.DEFAULT.LOCAL.QUEUE Select...

When this wizard completes, another wizard can be started automatically to create a matching object.

□ Start wizard to create a matching JMS Queue

Accept the defaults and click ***Finish***.

3. then you can modify the following line from the MQSample.java source and use the new queue 'Q1':

   ```
   // and define the name of the Queue
   private static final String qName = "Q1";
   ```

4. Save the changes to the file MQSample.java and then run it again as an application.
   This time the messages should look like this:

```
Connecting to queue manager: QM_80
Accessing queue: Q1
Sending a message...
...and getting the message back again
The message is: Hello, World!
Closing the queue
Disconnecting from the Queue Manager
Done!
```

Chapter 3: Importing and running a sample MQ JMS program

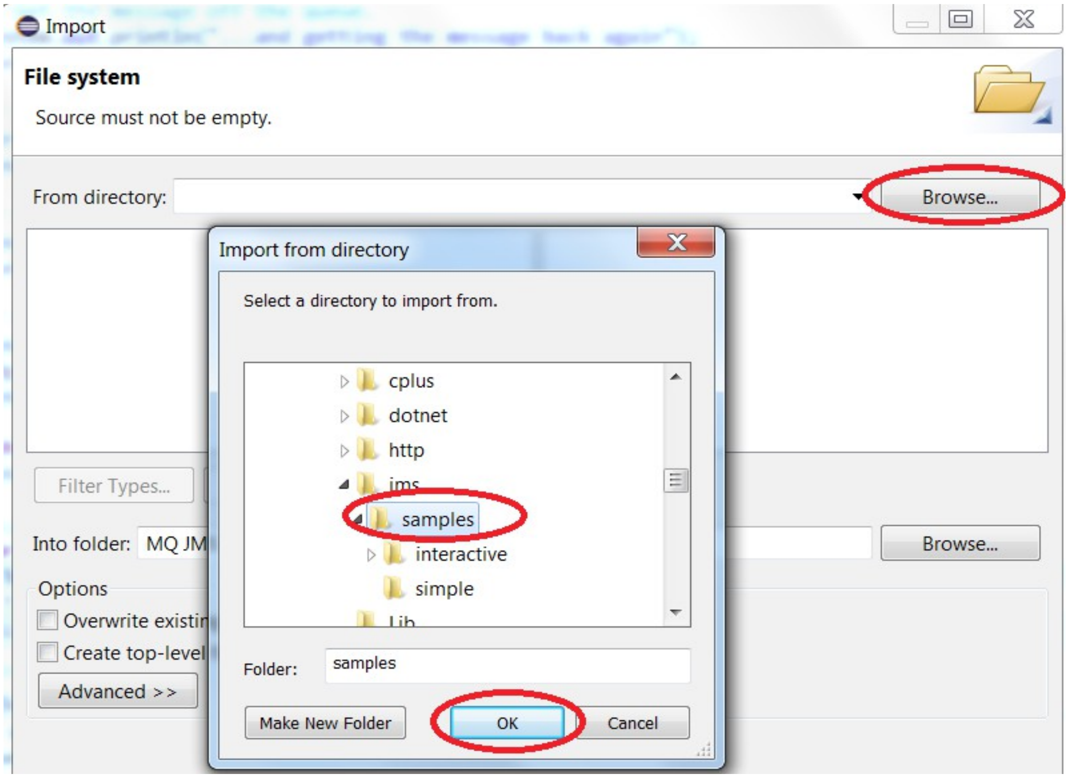
For this case, we will using another sample that requires to provide some parameters.

Before proceeding further, go to the ***MQ Explorer*** and create a queue manager called "***QM1***" and a local queue called "***Q1***". Go to section 2.1.2 and 2.1.3 for instructions.

1. Go back to ***Eclipse*** and follow the steps in Section 2 up to step 10.
   - Use "***MQ Jms Samples***" for the ***Project Name***.
   - When selecting the ***JAR files***, select the ones for ***JMS:***

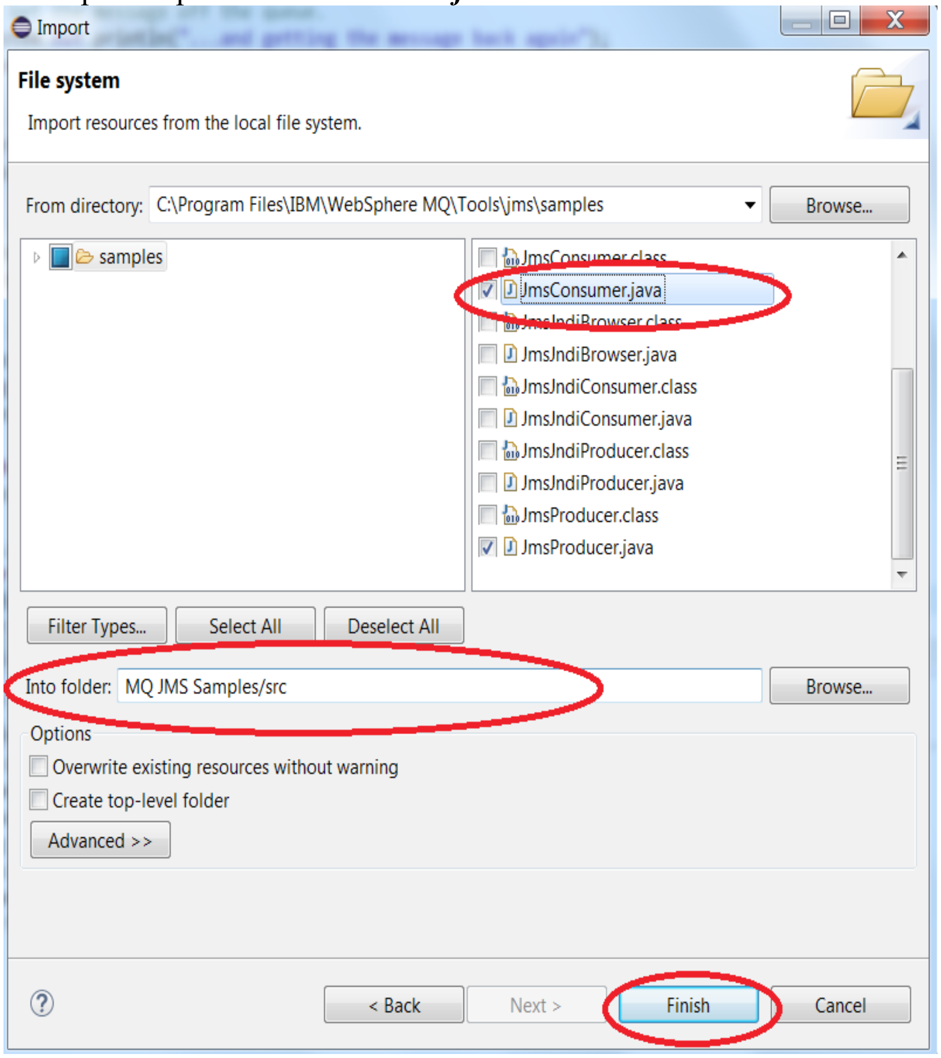
     com.ibm.mq.jms.Nojndi.jar
     com.ibm.mqjms.jar

2. Then Import the MQ JMS sample. In the "***File system***" panel, Click ***Browse...*** and browse to the path:
   C:\Program Files\IBM\Websphere MQ\tools\jms/samples
   Click ***OK***



3. In the left pane of the Import panel, click on "***samples***".
   In the right pane, select: ***JmsProducer.java***
   The "***Into folder***" field should contain "***MQ Jms Samples***".

4. Repeat the Import step for "***JmsConsumer.java***"



Import

File system

Import resources from the local file system.

From directory: C:\Program Files\IBM\WebSphere MQ\Tools\jms\samples

▷ ■ samples

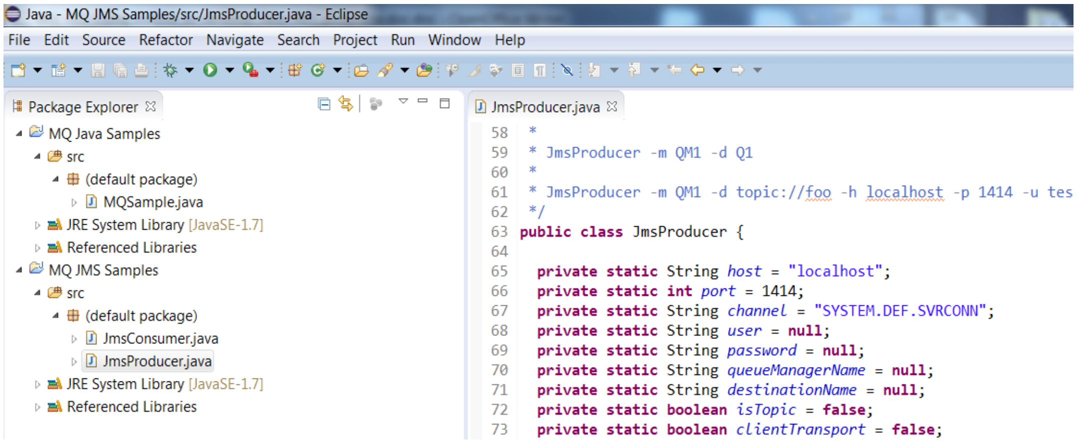
☐ JmsConsumer.class
☑ JmsConsumer.java
☐ JmsJndiBrowser.class
☐ JmsJndiBrowser.java
☐ JmsJndiConsumer.class
☐ JmsJndiConsumer.java
☐ JmsJndiProducer.class
☐ JmsJndiProducer.java
☐ JmsProducer.class
☑ JmsProducer.java

Into folder: MQ JMS Samples/src

Options
☐ Overwrite existing resources without warning
☐ Create top-level folder

Click ***Finish***.

5. From the ***Package Explorer***, expand “***MQ Jms Samples***”.
   Notice “(default package)”
   Expand it to see the Java source code.



```
 *
 * JmsProducer -m QM1 -d Q1
 *
 * JmsProducer -m QM1 -d topic://foo -h localhost -p 1414 -u tes
 */
public class JmsProducer {

  private static String host = "localhost";
  private static int port = 1414;
  private static String channel = "SYSTEM.DEF.SVRCONN";
  private static String user = null;
  private static String password = null;
  private static String queueManagerName = null;
  private static String destinationName = null;
  private static boolean isTopic = false;
  private static boolean clientTransport = false;
```

6. Open ***JmsProducer*** and ***JmsConsumer***. Study the classes.
   If needed, modify the attributes of either program.

   The following attributes are changeable by editing the source code or by providing it as parameters.

   - host
   - port
   - channel
   - user
   - password

   The following attributes are changeable only by providing it as parameters:

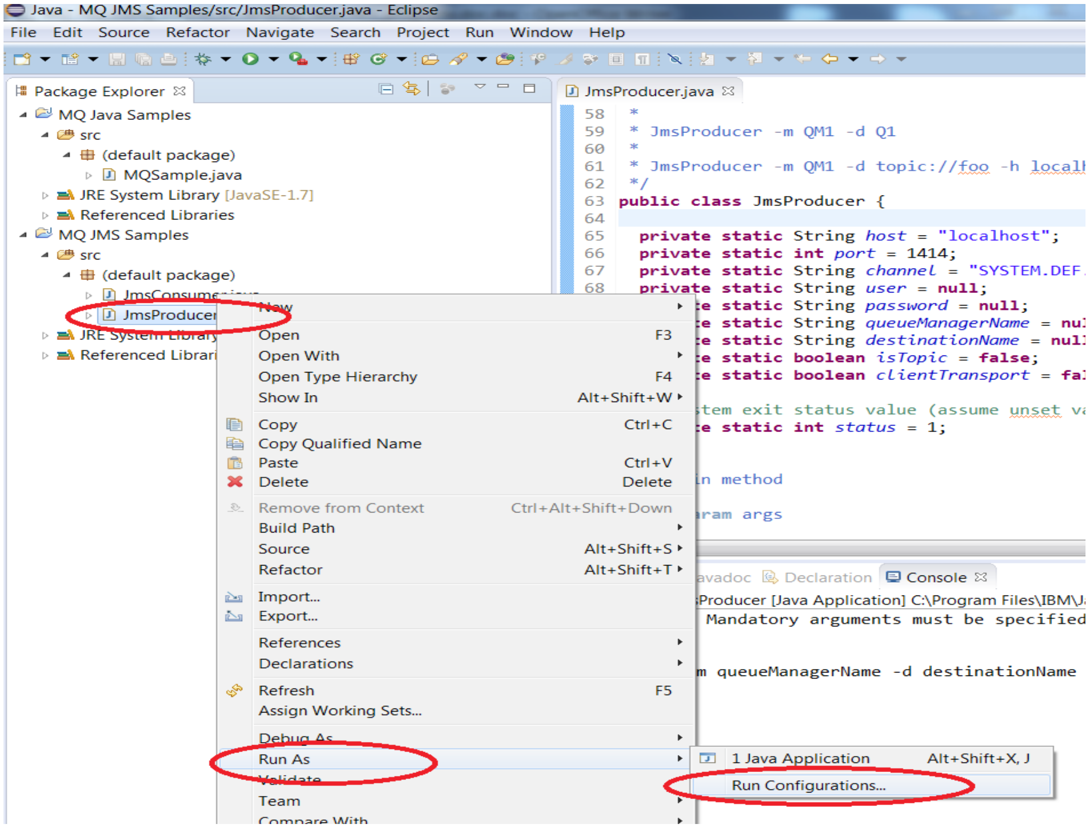
   - queueManagerName
   - destinationName

Section 3.1: Provide parameters or modify the source code for the sample to run in your environment.

Provide the parameter, as shown in the steps below to specify a queue manager name and destination.

> Note: The following steps are shown for the ***JmsProducer*** class, but they can also be used for the ***JmsConsumer*** class.

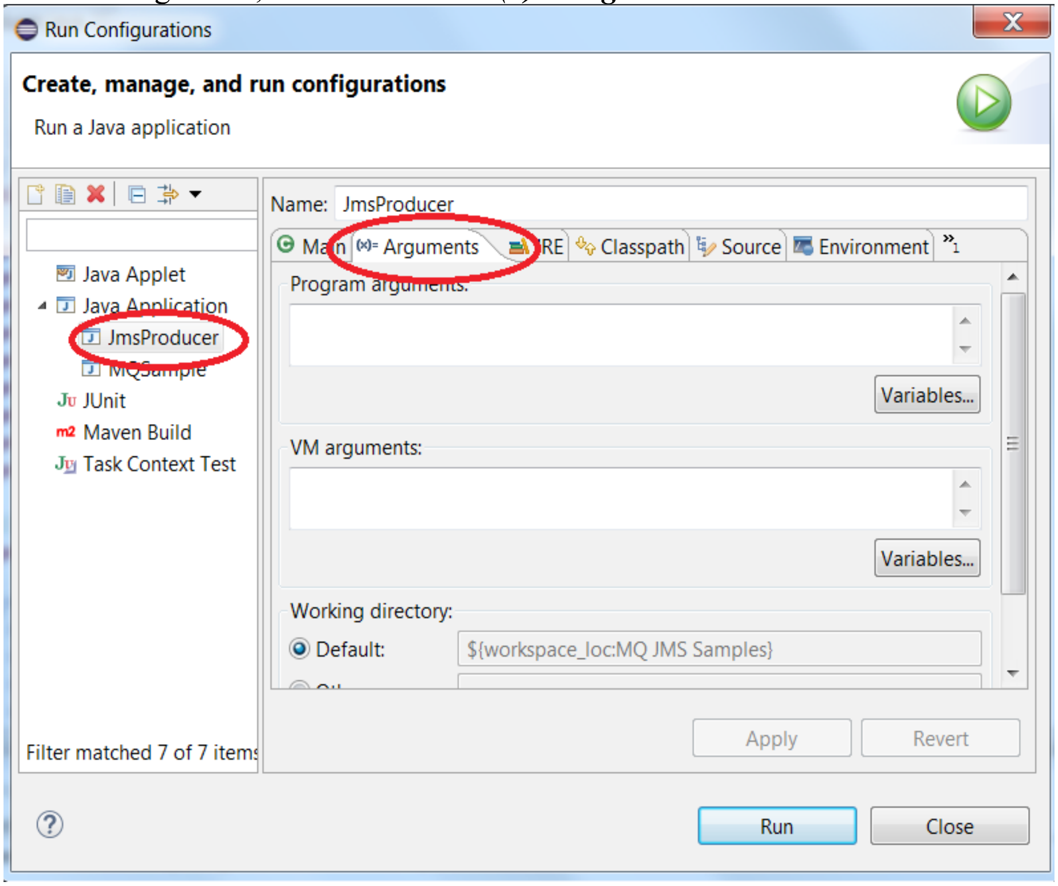
1. In the workspace, right-click ***Run As*** → ***Run Configurations...***



```
 *
 * JmsProducer -m QM1 -d Q1
 *
 * JmsProducer -m QM1 -d topic://foo -h locall
 */
public class JmsProducer {

  private static String host = "localhost";
  private static int port = 1414;
  private static String channel = "SYSTEM.DEF
  private static String user = null;
e static String password = null;
e static String queueManagerName = nul
e static String destinationName = nul
e static boolean isTopic = false;
e static boolean clientTransport = fal

tem exit status value (assume unset va
e static int status = 1;

n method

ram args
```

```
Producer [Java Application] C:\Program Files\IBM\J
 Mandatory arguments must be specified

m queueManagerName -d destinationName
```

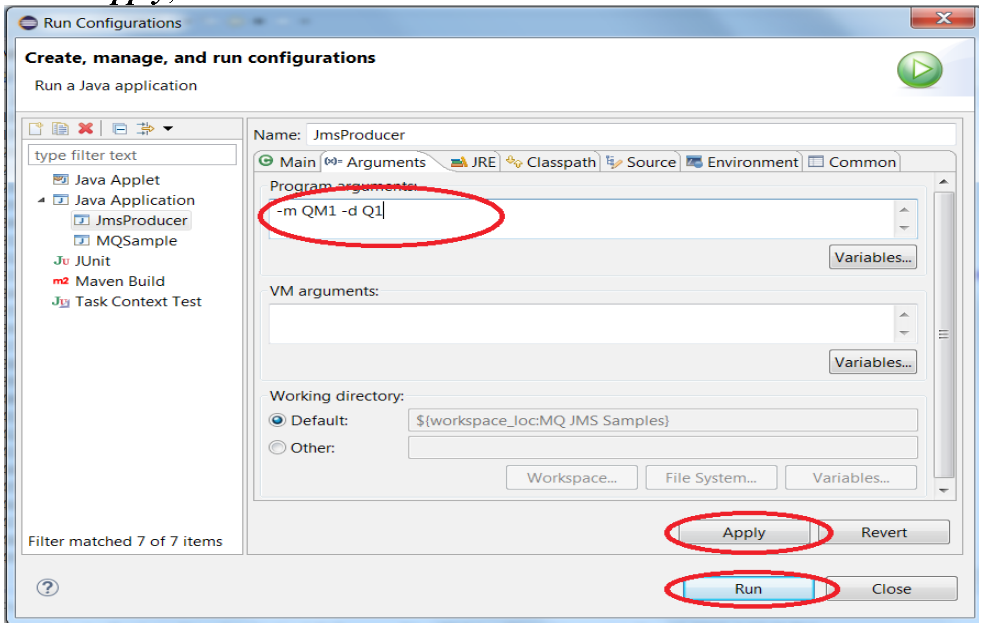
2. In the window “***Create, Manage and Run configurations***”, notice that in the left panel the application “***JmsProducer***” is already selected.
In the right side, click on the tab: “***(x) = Arguments***”



Name: JmsProducer

● Default: ${workspace_loc:MQ JMS Samples}

3. In the "***Program arguments***" field type: ***-m QM1 -d Q1***
   Click ***Apply***, then ***Run***.



Run Configurations

Create, manage, and run configurations

Run a Java application

Name: JmsProducer

Program arguments:
-m QM1 -d Q1

VM arguments:

Working directory:
● Default: ${workspace_loc:MQ JMS Samples}
○ Other:

Java Applet
Java Application
JmsProducer
MQSample
JUnit
Maven Build
Task Context Test

Filter matched 7 of 7 items

4. If you get a reason code 2058, check if the QM1 was created correctly.
In the "Console" tab you will see a prompt:
   Enter some text to be sent in a message <ENTER to finish>:

In this case, the following text was entered:
      message-1

You will see:

```
Sent message:
  JMSMessage class: jms_text
  JMSType:          null
  JMSDeliveryMode:  2
  JMSDeliveryDelay: 0
  JMSDeliveryTime:  1434740621461
  JMSExpiration:    0
  JMSPriority:      4
  JMSMessageID:     ID:414d5120514d5f38302020202020202020206a61845520002402
  JMSTimestamp:     1434740621461
  JMSCorrelationID: null
  JMSDestination:   queue:///Q1
  JMSReplyTo:       null
  JMSRedelivered:   false
    JMSXAppID: les\IBM\Java70\bin\javaw.exe
    JMSXDeliveryCount: 0
    JMSXUserID: rivera
```

```
   JMS_IBM_PutApplType: 11
   JMS_IBM_PutDate: 20150619
   JMS_IBM_PutTime: 19034150
message-1
Enter some text to be sent in a message <ENTER to finish>:
(note: just the ENTER key was pressed, in order to exit)
SUCCESS
```

+++ end +++