IBM Software
WebSphere

WebSphere Lab Jam

Connectivity

WebSphere MQ

Lab Exercises

An IBM Proof of Technology

Catalog Number

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

Lab 1 Introduction to WebSphere MQ

In this lab you will be introduced to the WebSphere® MQ Explorer, the primary interface for administering a WebSphere MQ environment. You will have an opportunity to see how easy it is to create various MQ objects, view their status and manipulate them.

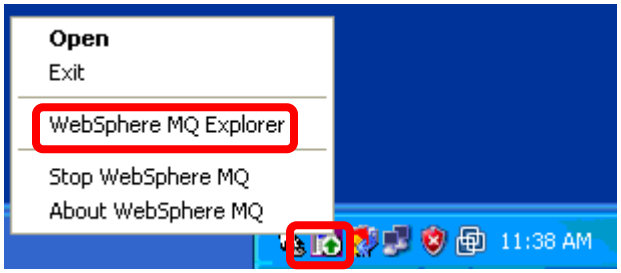
You will also exercise some basic command line facilities that will allow you to place messages onto queues and remove messages from queues.

1.1 Create a Queue Manager

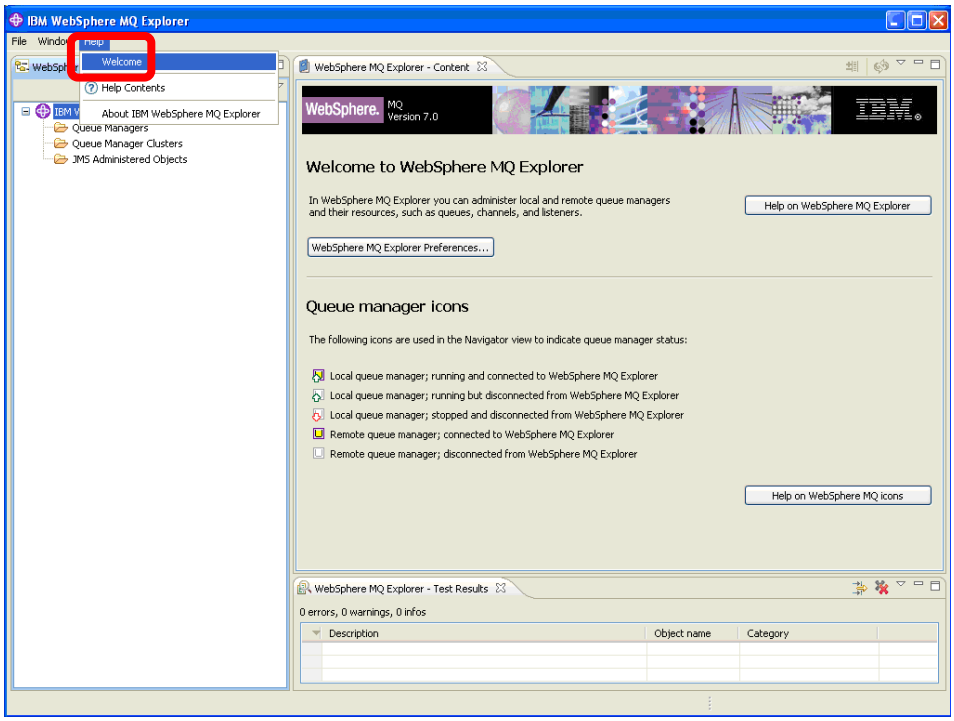
Before you can do any useful work in a WebSphere MQ environment you must have a queue manager present.

__1. The indicated icon in the lower right hand corner represents WebSphere MQ on this system.

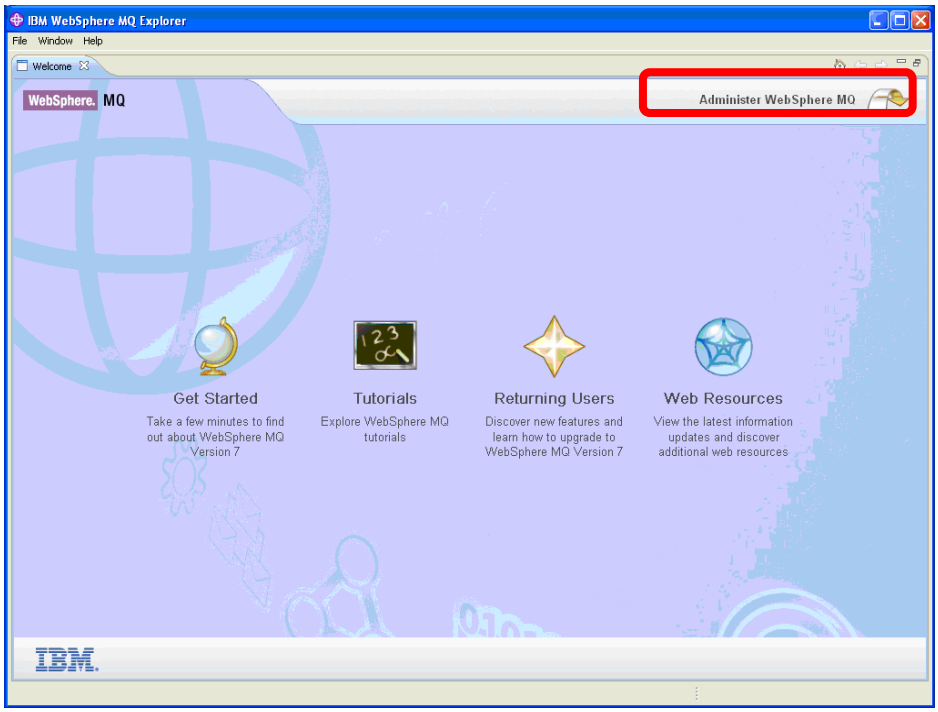
__2. Start the MQ Explorer by **right-clicking** on the icon and selecting **WebSphere MQ Explorer**



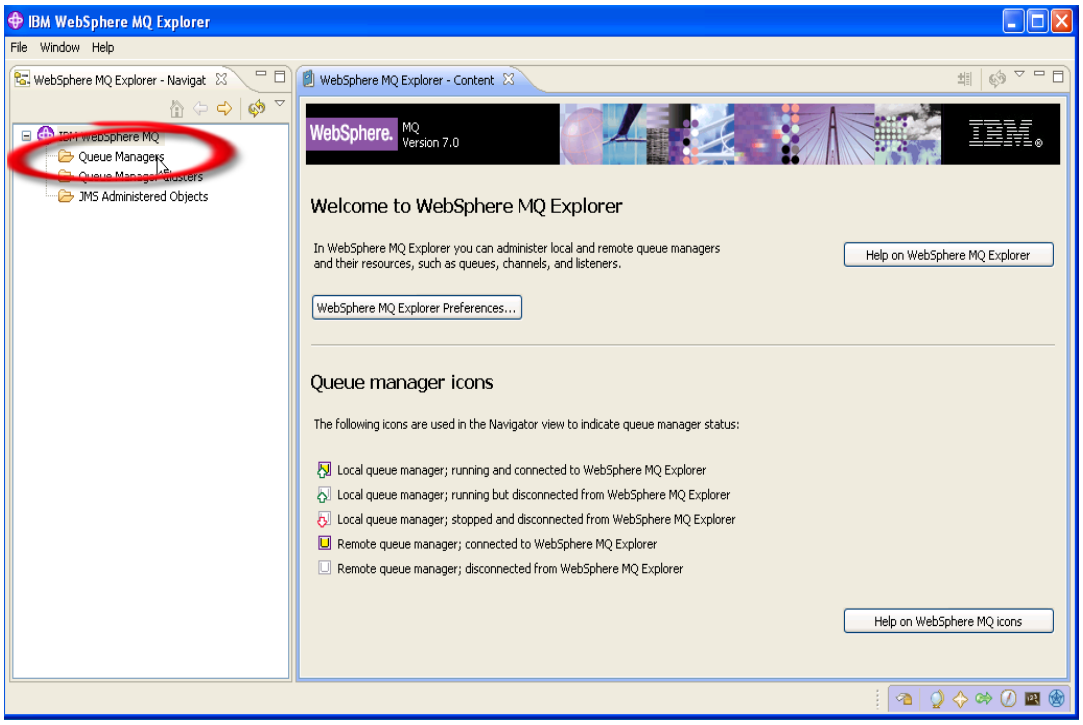
__3. The welcome screen provides a nice selection of resources for the product. On the Menu Bar, click on **Help** and select **Welcome** to bring it into view



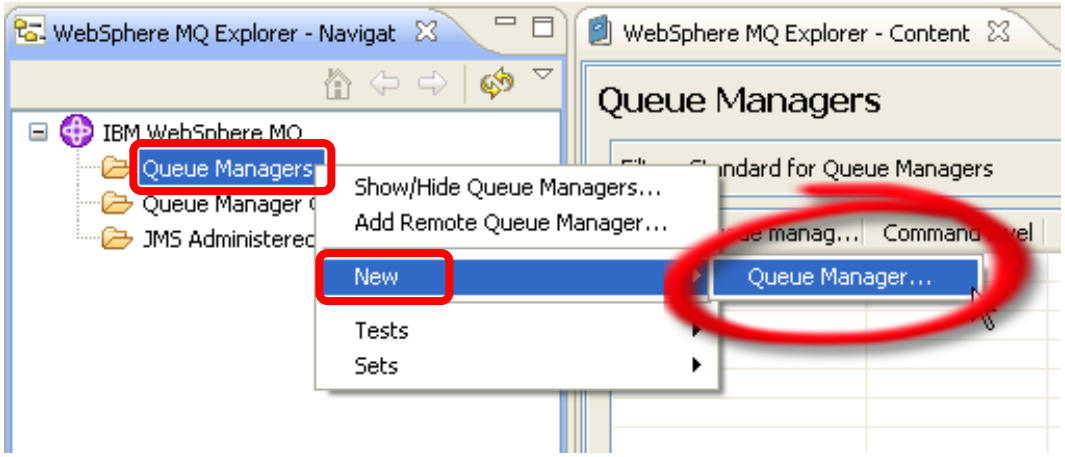
__4. Note the various options on the Welcome screen and explore them if you would like. Then return to the MQ Explorer workbench by clicking the **Administer WebSphere MQ** icon in the upper right-hand corner. The first time you launch MQ Explorer after an install of WebSphere MQ this Welcome screen will be displayed automatically.



__5. The left-hand pane in MQ Explorer is called the **Navigation pane**. In the navigation pane, locate the **Queue Managers** folder.



__6. Right-click on **Queue Managers** and select **New** then **Queue Manager…**



__7. Create a new queue manager and name it **WMQ7**. Check the box to make this the **default queue manager**. Note that the use of a default queue manager is not recommended in a production environment as it allows requests from a program or command to complete successfully without the inclusion of a queue manager name. We are using this option here in the labs as a convenience to reduce typing for you.

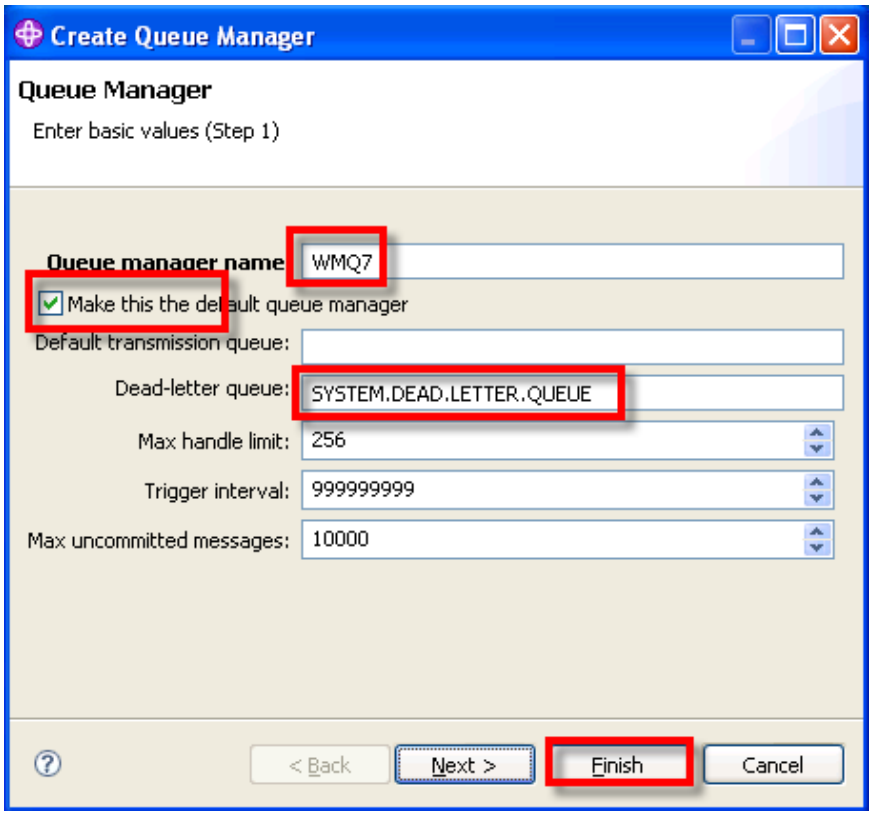
****

Important!!!!! Be sure to check the box that indicates this is the default queue manager!!!

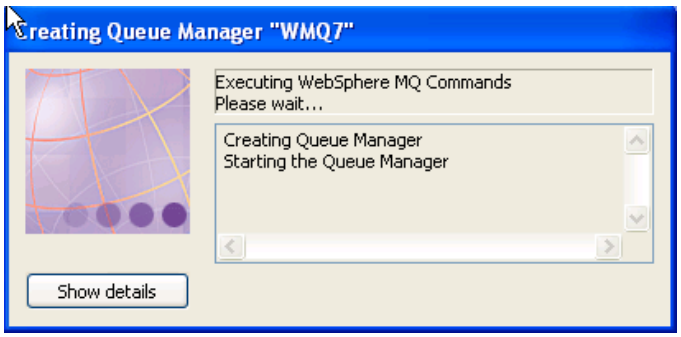
If you do not do this you will have issues in future labs!!!!!!

****

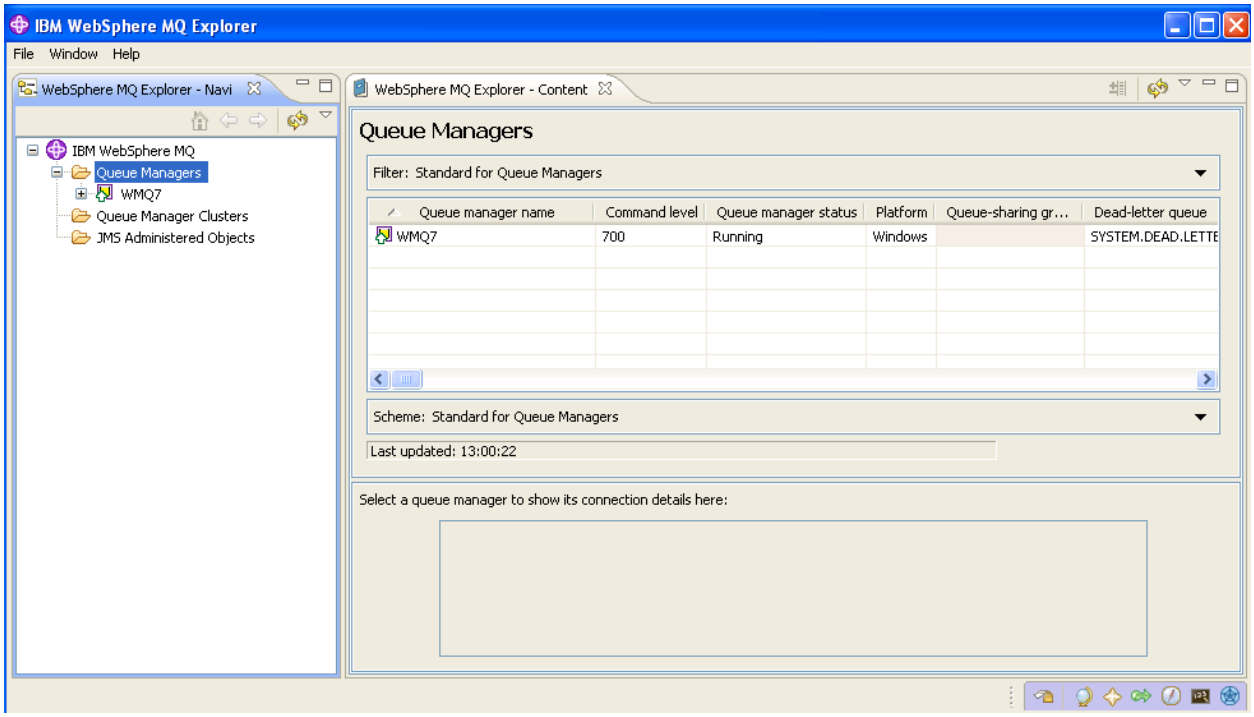
Specify **SYSTEM.DEAD.LETTER.QUEUE** as the **dead letter queue**, then click the **Finish** button.



__8. The create queue manager process takes a few seconds during which time the following panel is displayed…



__9. The newly created queue manager is displayed in MQ Explorer. In the Content pane on the right-hand side you can see details about the queue manager, such as its status, the name of the dead letter queue, etc.



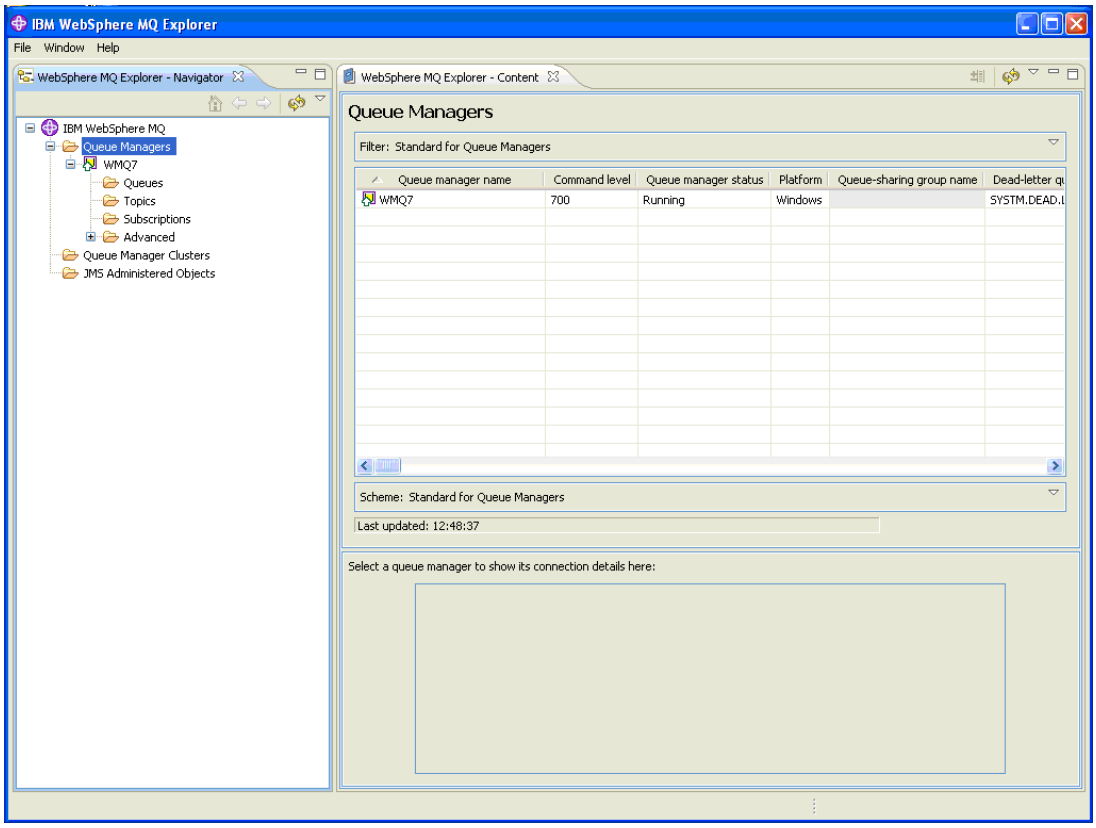
This concludes this portion of Lab 1.

1.2 Testing Basic Functionality

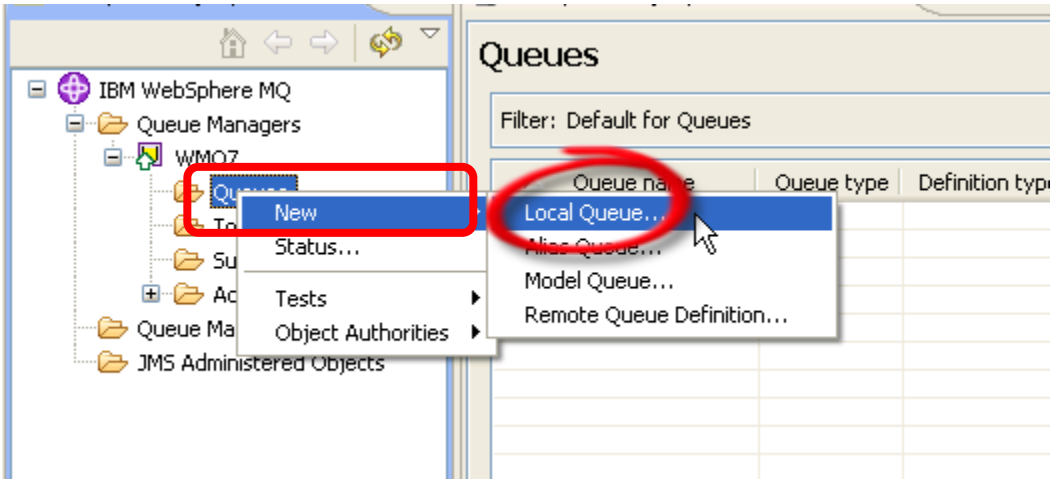
In this section of the lab you will create a local queue, place a test message in the queue, browse the message, clear it from the queue and finally delete the queue.

__1. Expand the new **WMQ7** queue manager by clicking on the **"+"** symbol in front of it in the navigation tree.

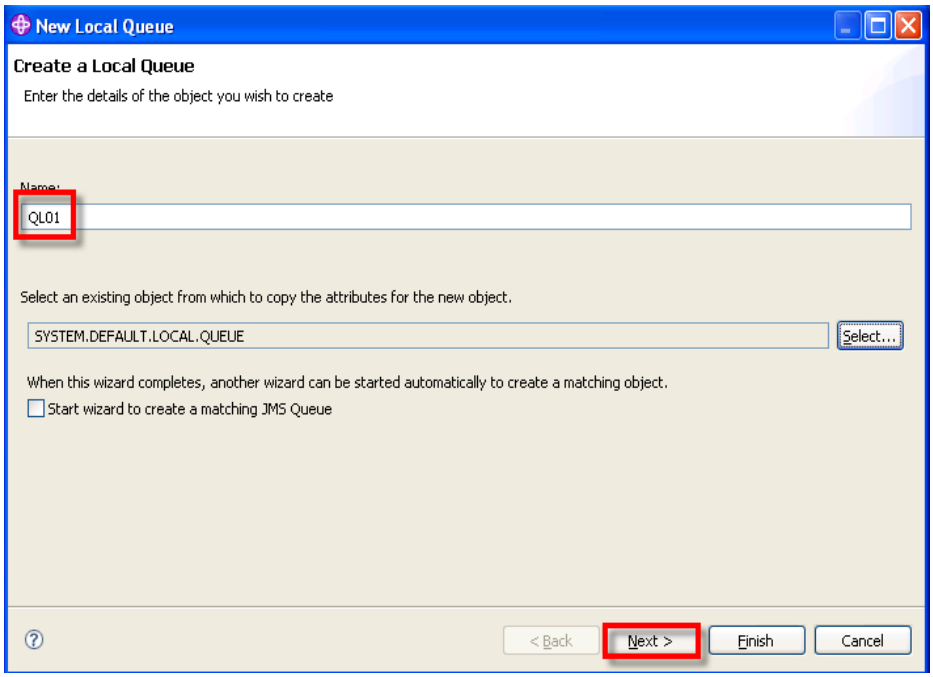
__2. Observe the tree structure that is displayed in the Navigator pane. There are separate folders for *Queues*, *Topics* and *Subscriptions*, as well as one labeled *Advanced*.



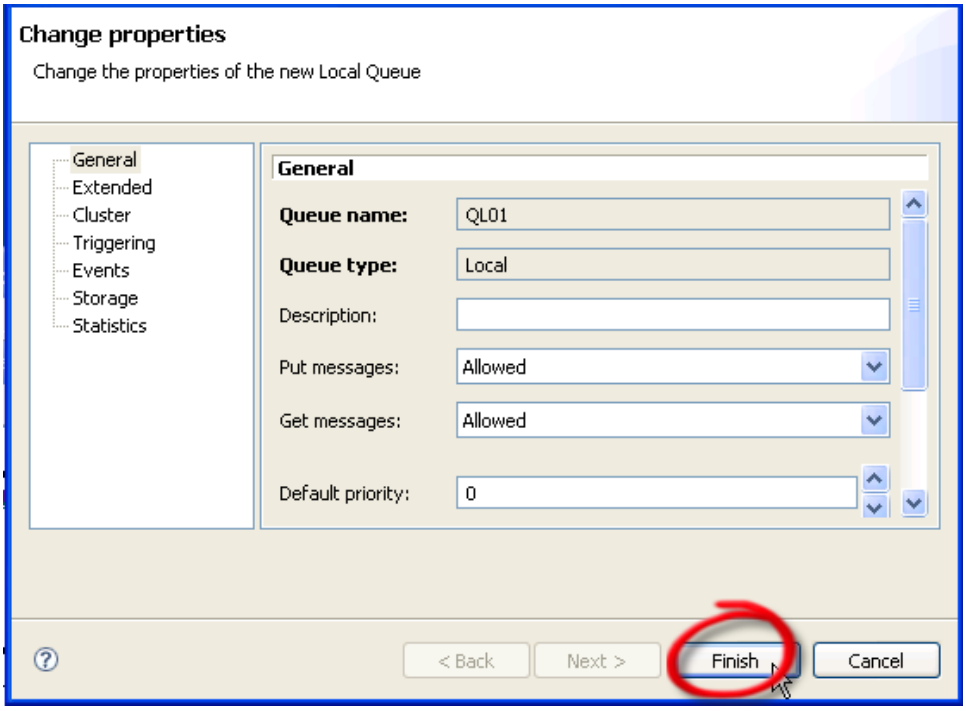
__3. Now you will create a new queue. Within the WMQ7 queue manager **right-click** on **Queues** then select **New** then **Local Queue**.



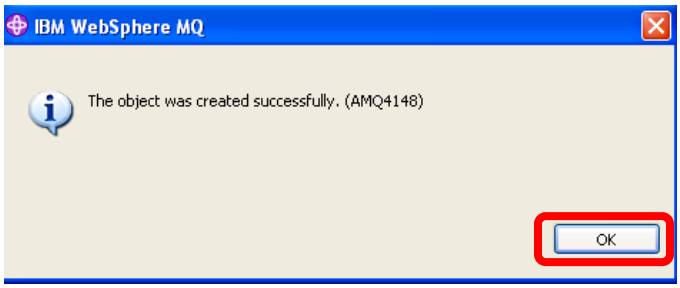
__4. Name the new queue **QL01** then click on the **Next** button.



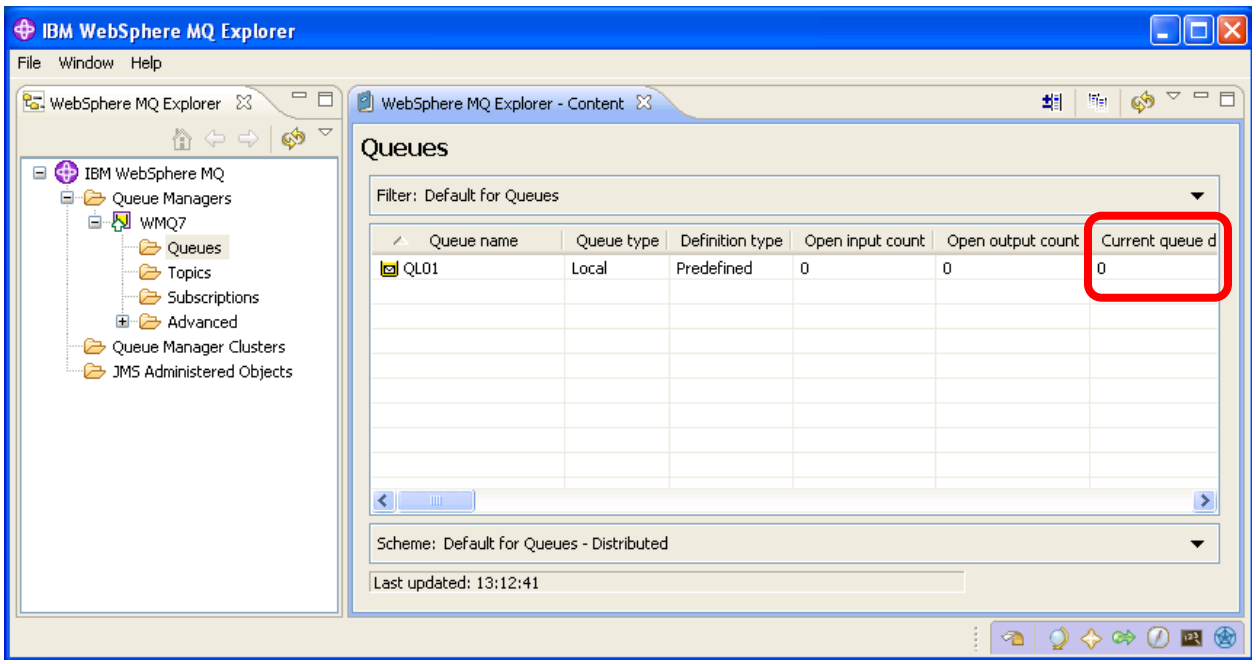
__5. Note the variety of tabs on the left. There are many characteristics or properties for a queue. Explore some of them if so desired. In most cases you can utilize the defaults. For purposes of this lab you can accept all of the defaults – click **Finish** to create the queue.



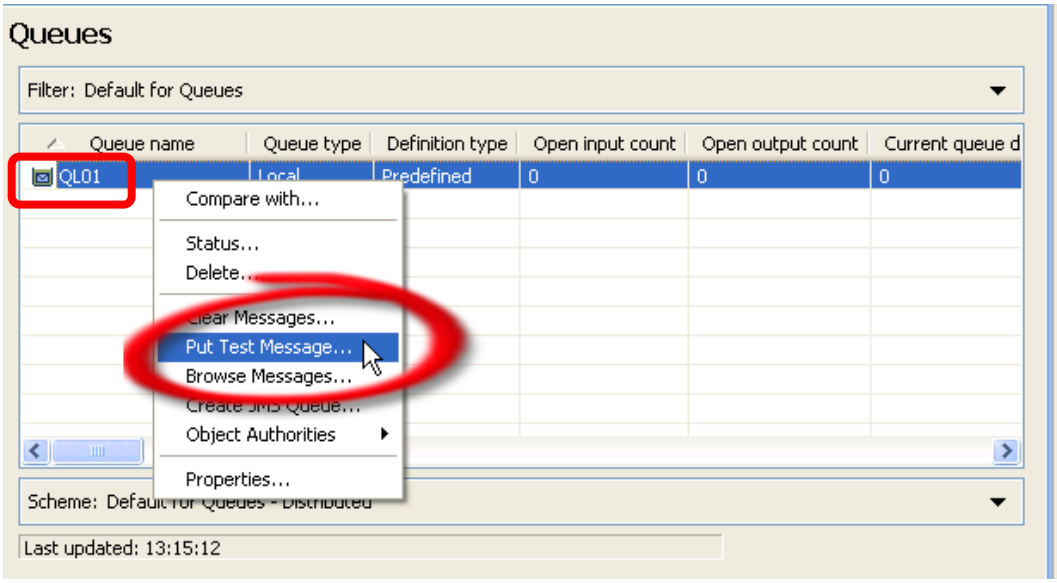
__6. Dismiss the completion panel by clicking on the **OK** button.



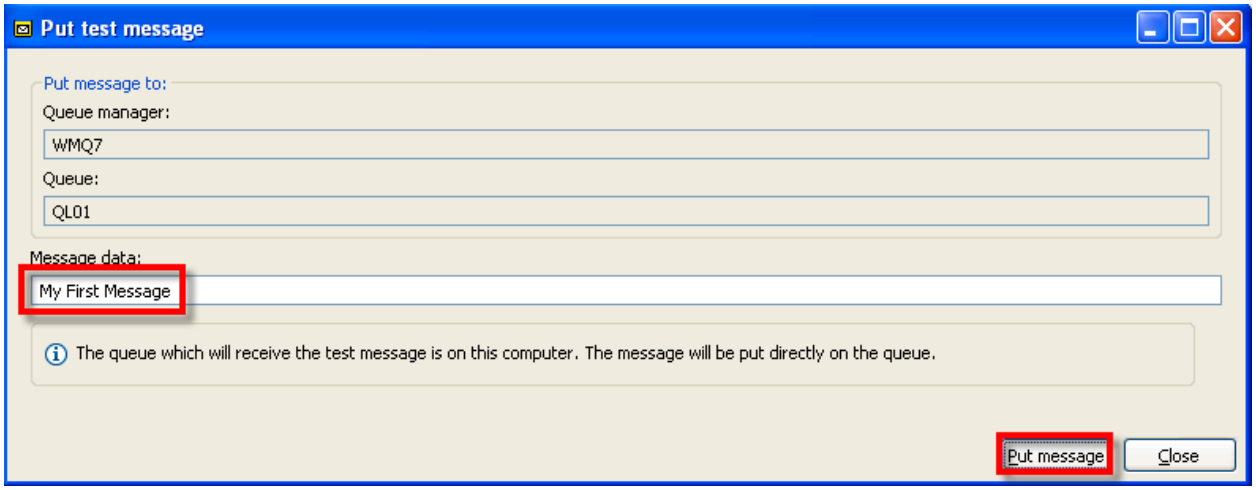
__7. In the Content pane on the right-hand side the newly created queue is displayed along with its various properties. The **Current queue depth** property is a measure of how many messages are currently in the queue. Note that the current queue depth is 0 as you would expect since you just created the queue.



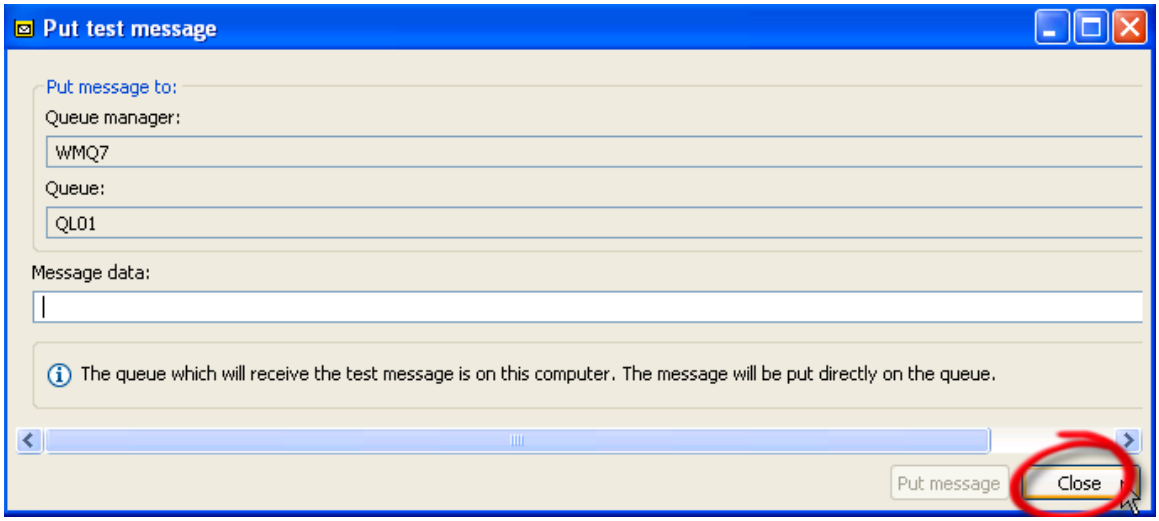
__8. Next you will place a message in the queue. **Right-click** on the **QL01** queue and select **Put Test Message**.



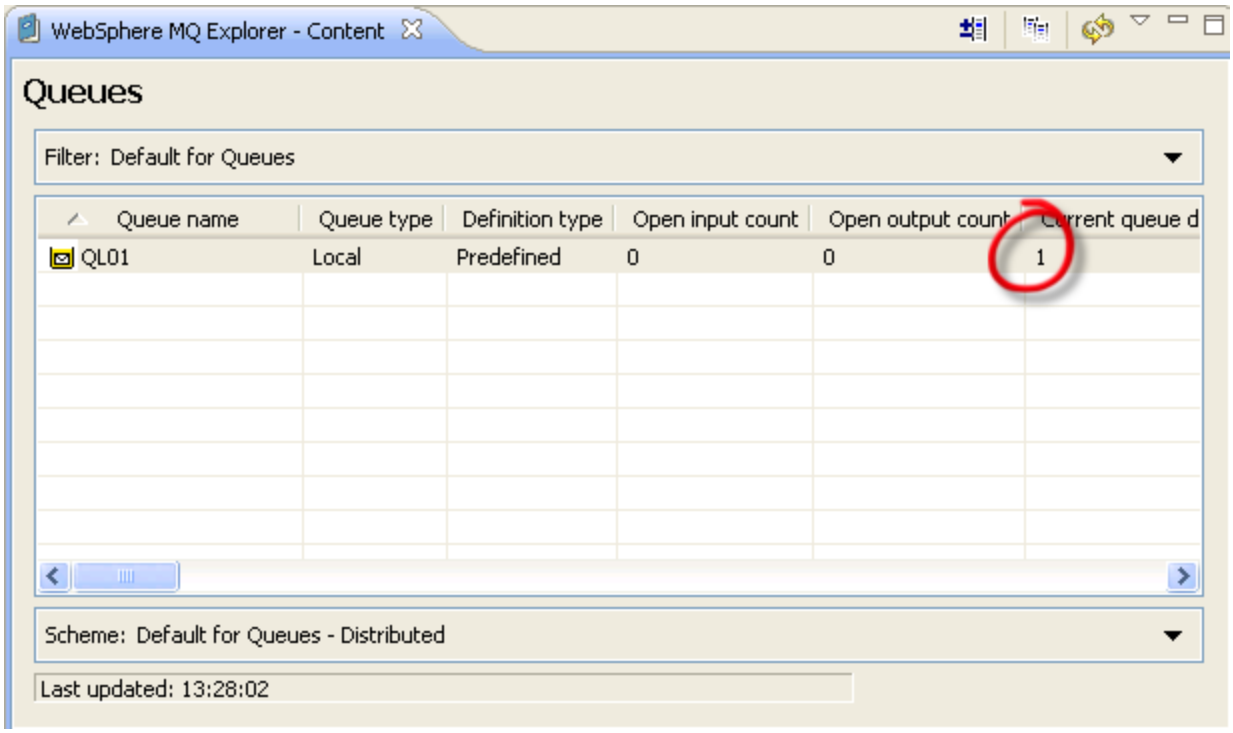
__9. Enter a test message by typing something of your choosing in the **Message data** field and click on the **Put message** button.



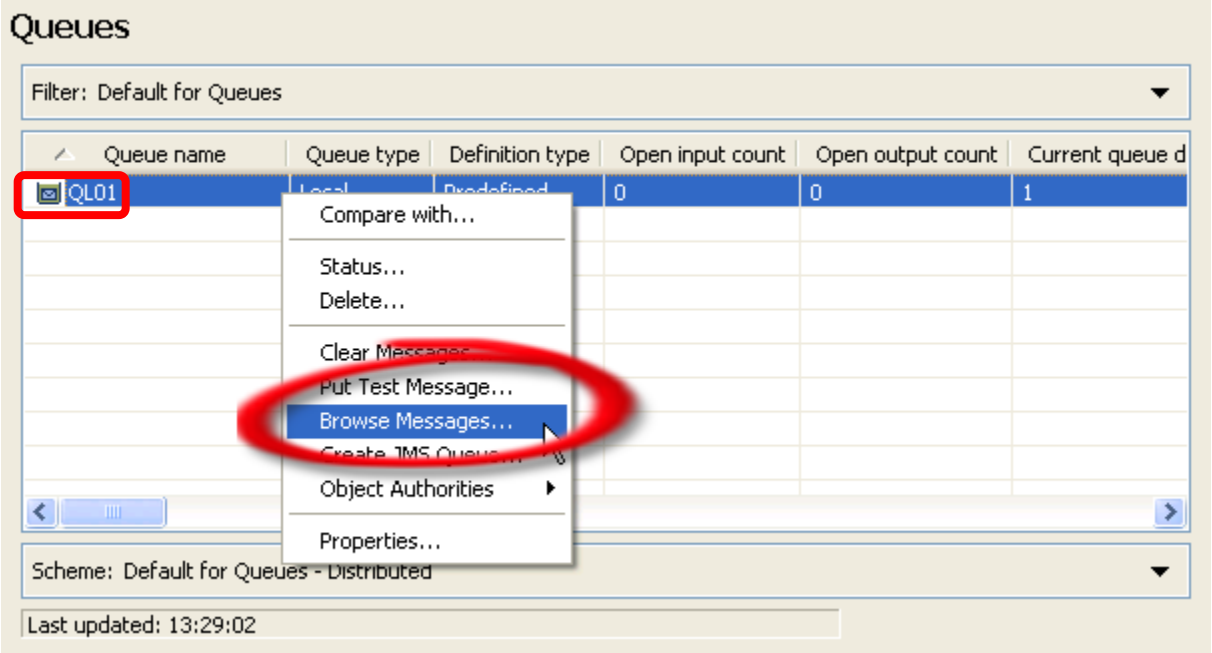
__10. Close the Put test message panel by clicking on the **Close** button.



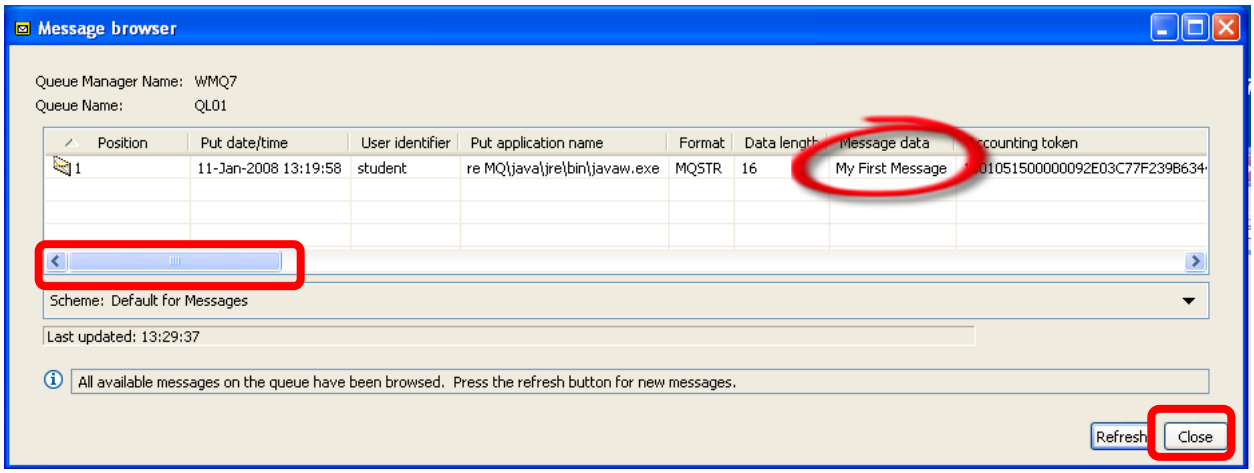
When you return to the display of queues, note that the **Current queue depth count is now 1**. This represents the test message you just placed into the queue.



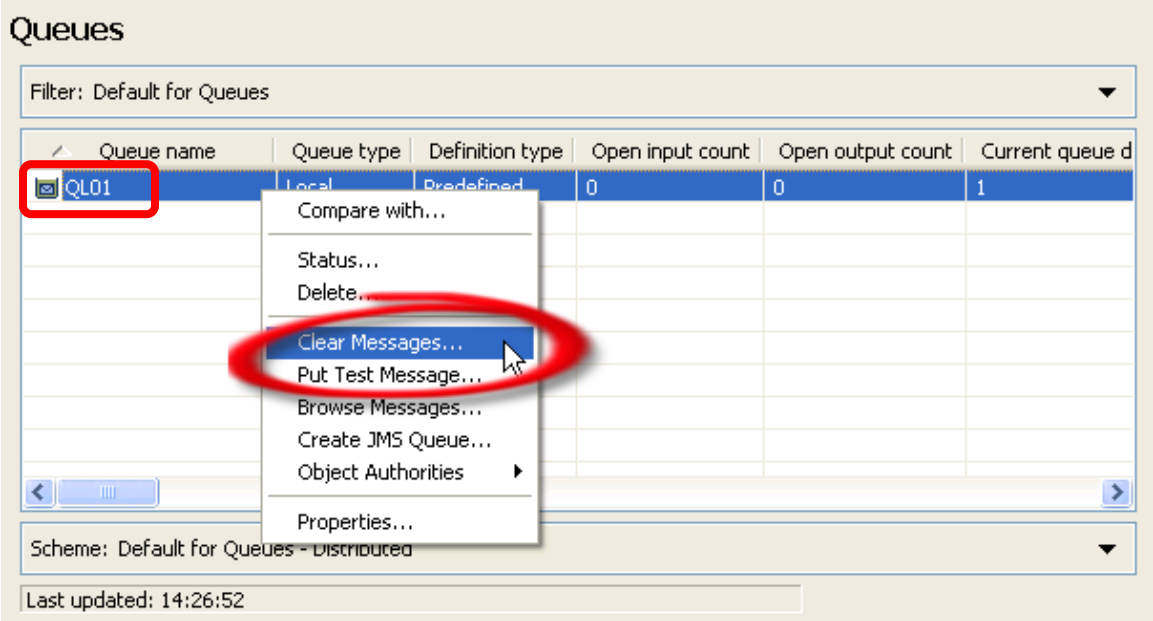
__11. You can view details about the messages on a queue – **right-click** on queue **QL01** and select **Browse Messages**.



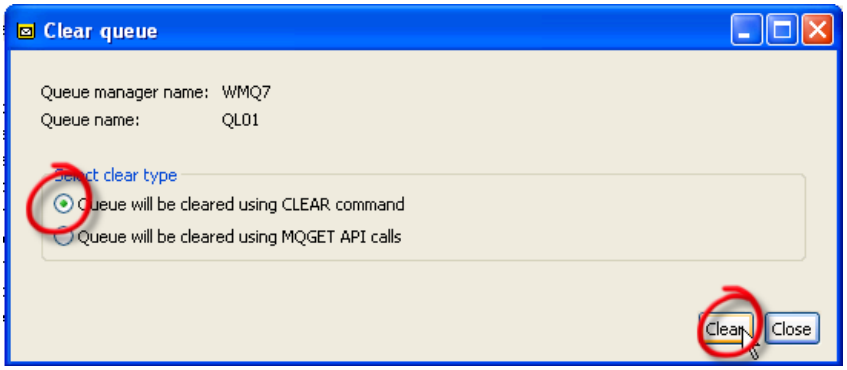
__12. Here is the resulting display. Note that there is a **scroll bar** at the bottom that allows you to view additional data that is not currently in view. There will be one such line displayed for each message in the queue. Click on the **Close** button.



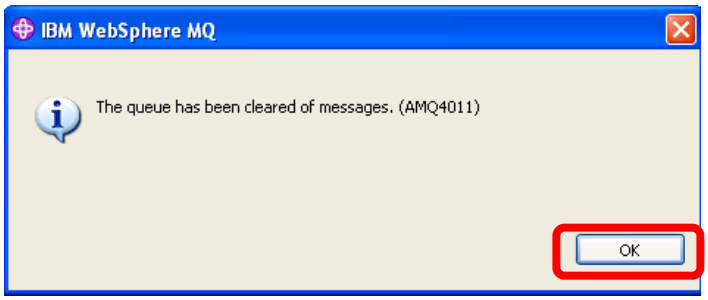
__13. You will now delete your test message from the queue. **Right-click** on queue **QL01** and select **Clear Messages**…..note that this action will remove <u>all</u> messages from the queue.



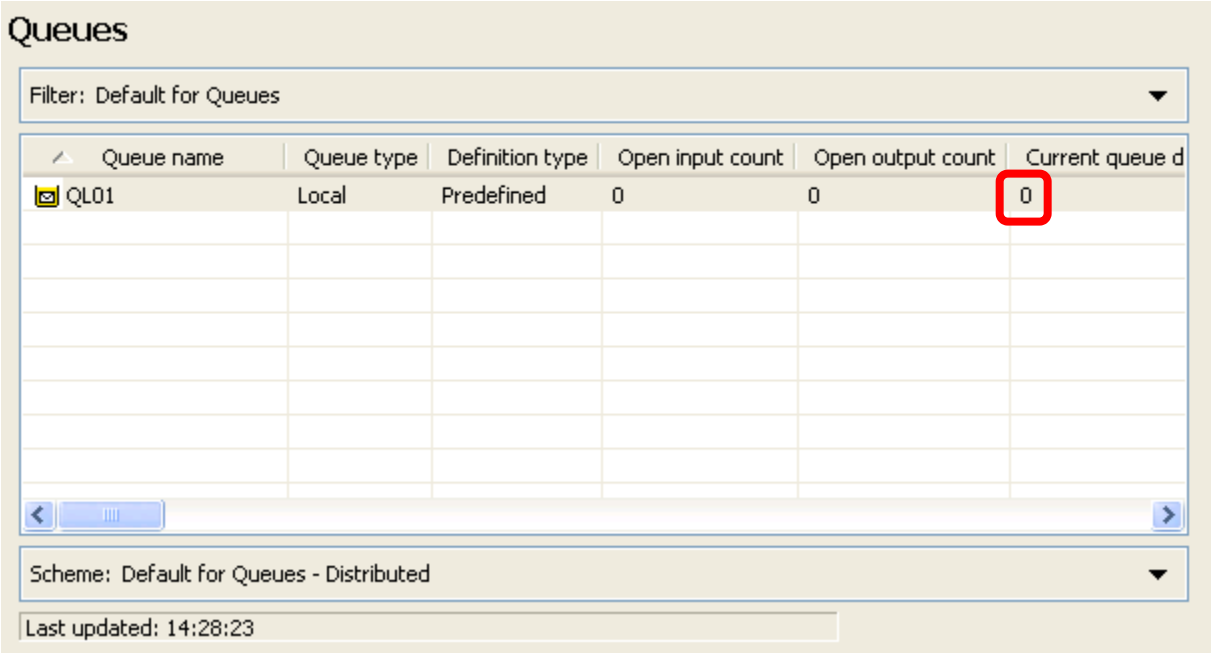
__14. Accept the default selection of using the **CLEAR command**. This option cannot be used if any application has the queue open. In that case the second option would be required. Click on the **Clear** button.



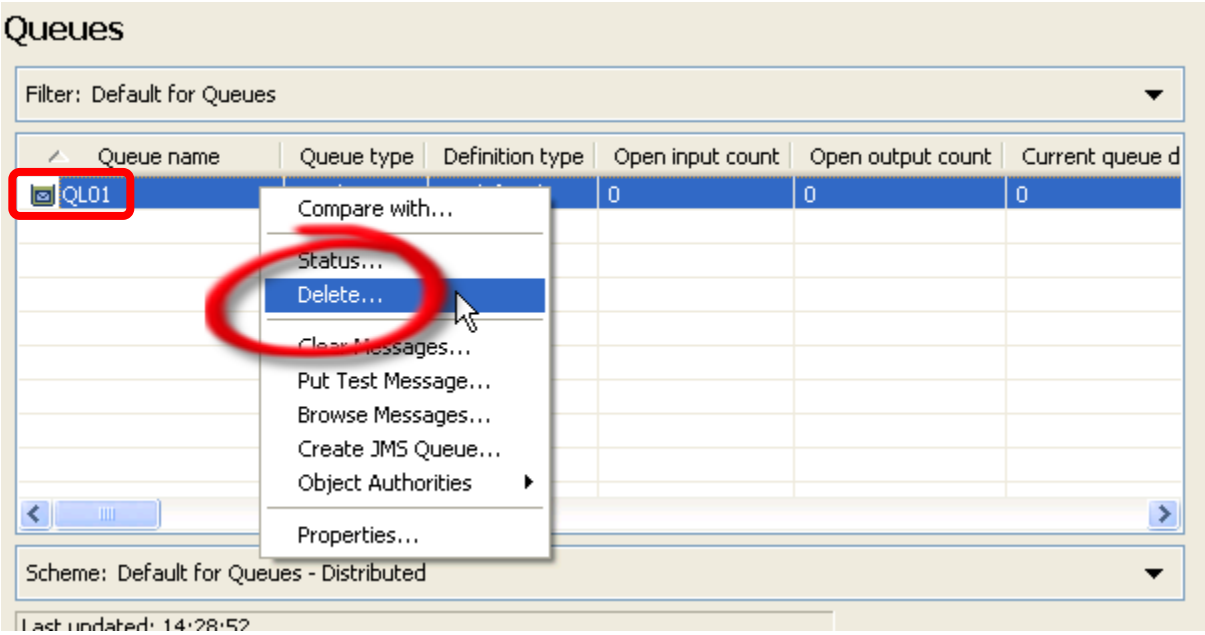
__15. Click on the **OK** button to dismiss the confirmation panel.



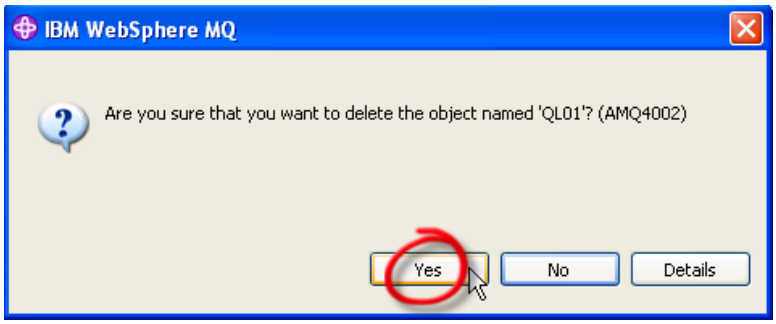
You can now see that the queue is indeed empty.



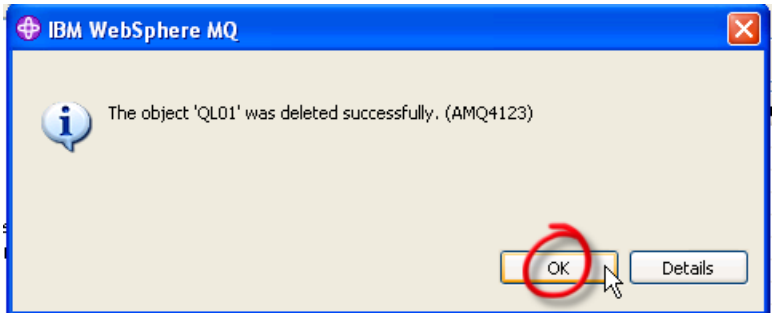
__16. You will now delete the queue. **Right-click** on the **QL01** queue and select **Delete…**



__17. Confirm the request by clicking on the **Yes** button.



__18. Dismiss the confirmation panel by clicking on the **OK** button.



This concludes this portion of Lab 1.

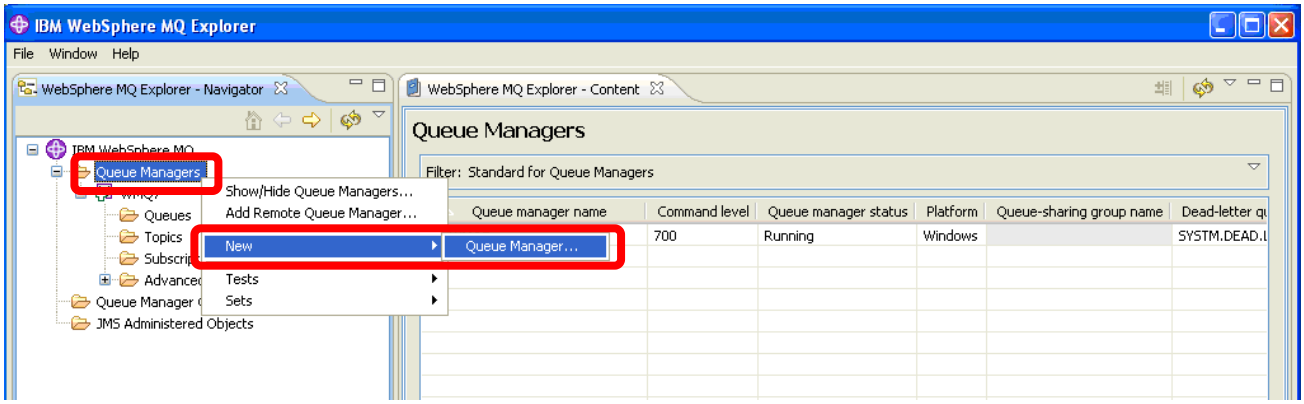
1.3 Grouping Queue Managers

In this portion of the lab you will use a feature of MQ Explorer that allows grouping of queue managers into Sets. This offers a convenient way to view and/or manage a subset of queue managers. A particular queue manager may be a member of one Set or many Sets or no Set at all. The only restriction is that a Set may not contain another Set.

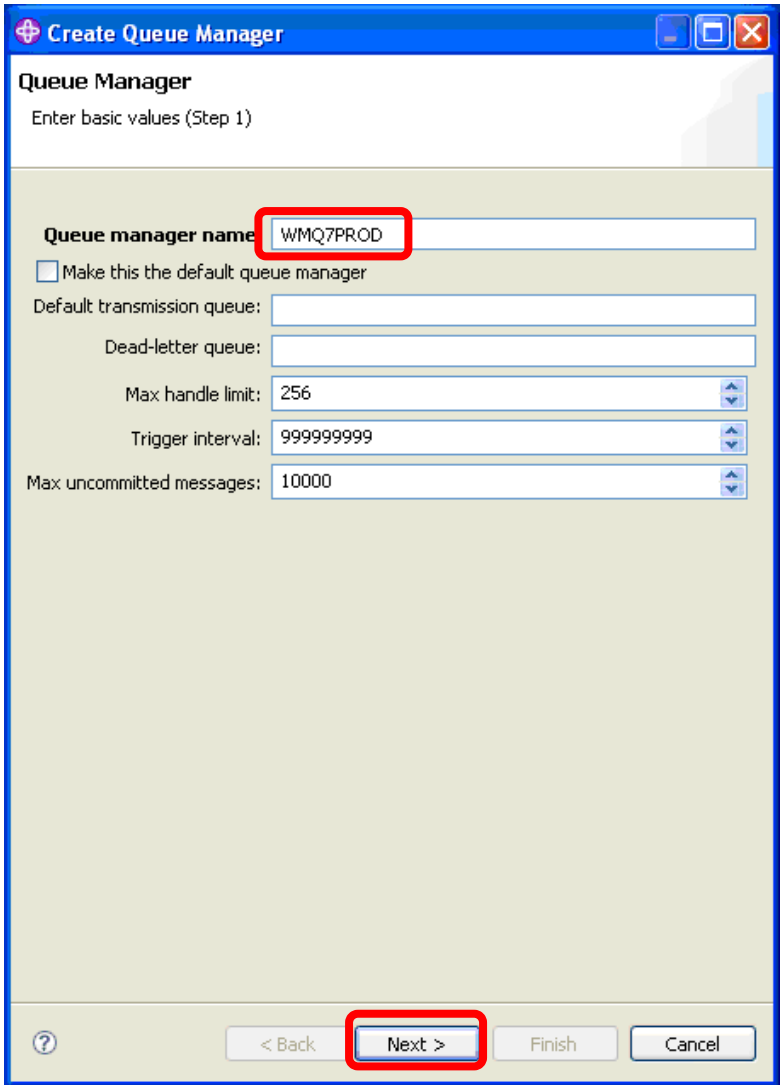
Using the MQ Explorer, you will now create the following queue manager and assign it the indicated port number for the listener:

WMQ7PROD → port number = 1415

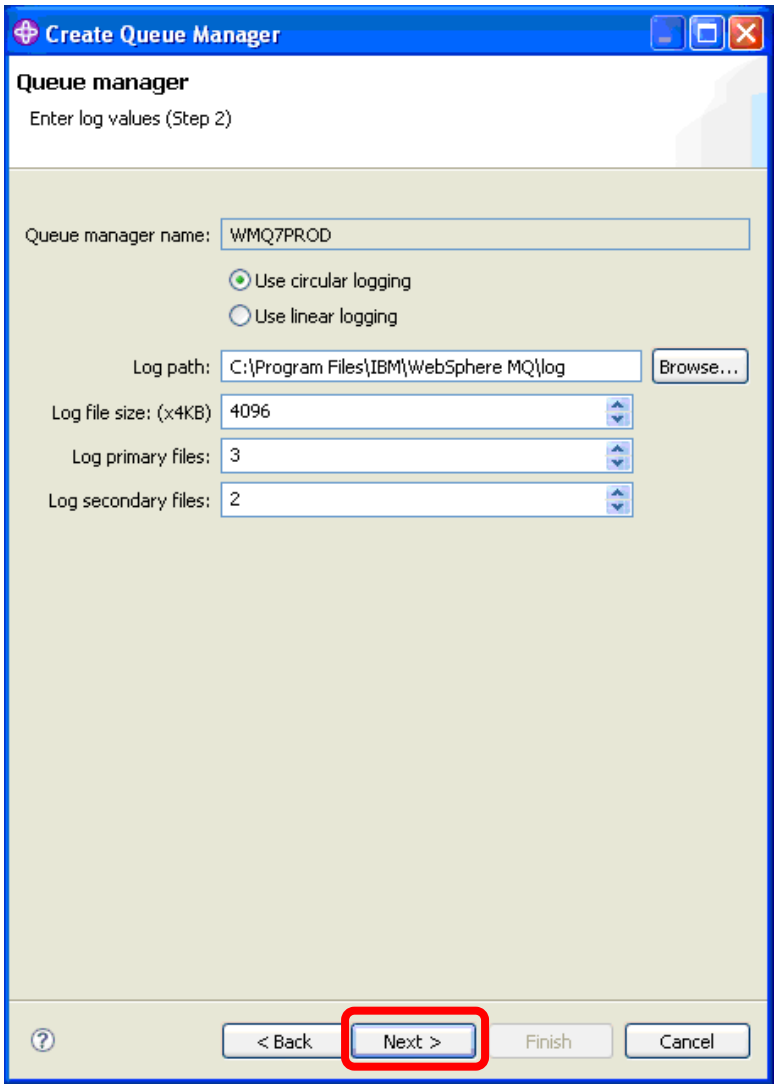
__1. Right-click on **Queue Managers** and select **New**, then **Queue Manager**



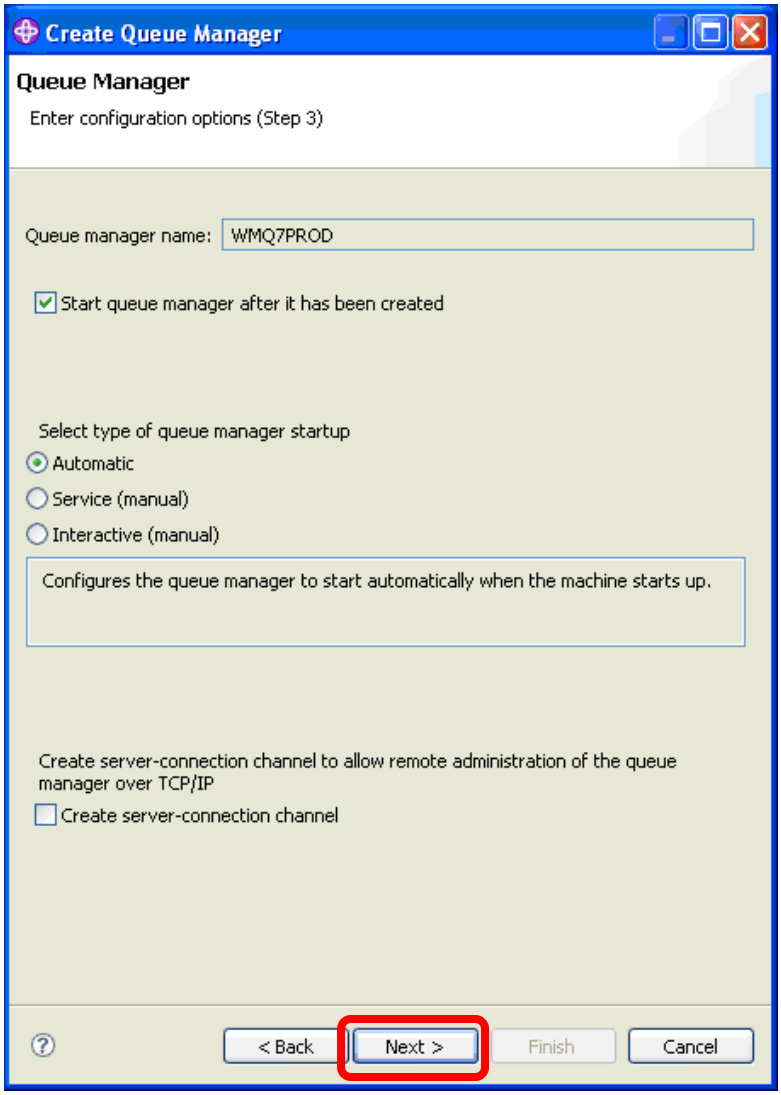
__2. Enter **WMQ7PROD** as the name and click the **Next** button.



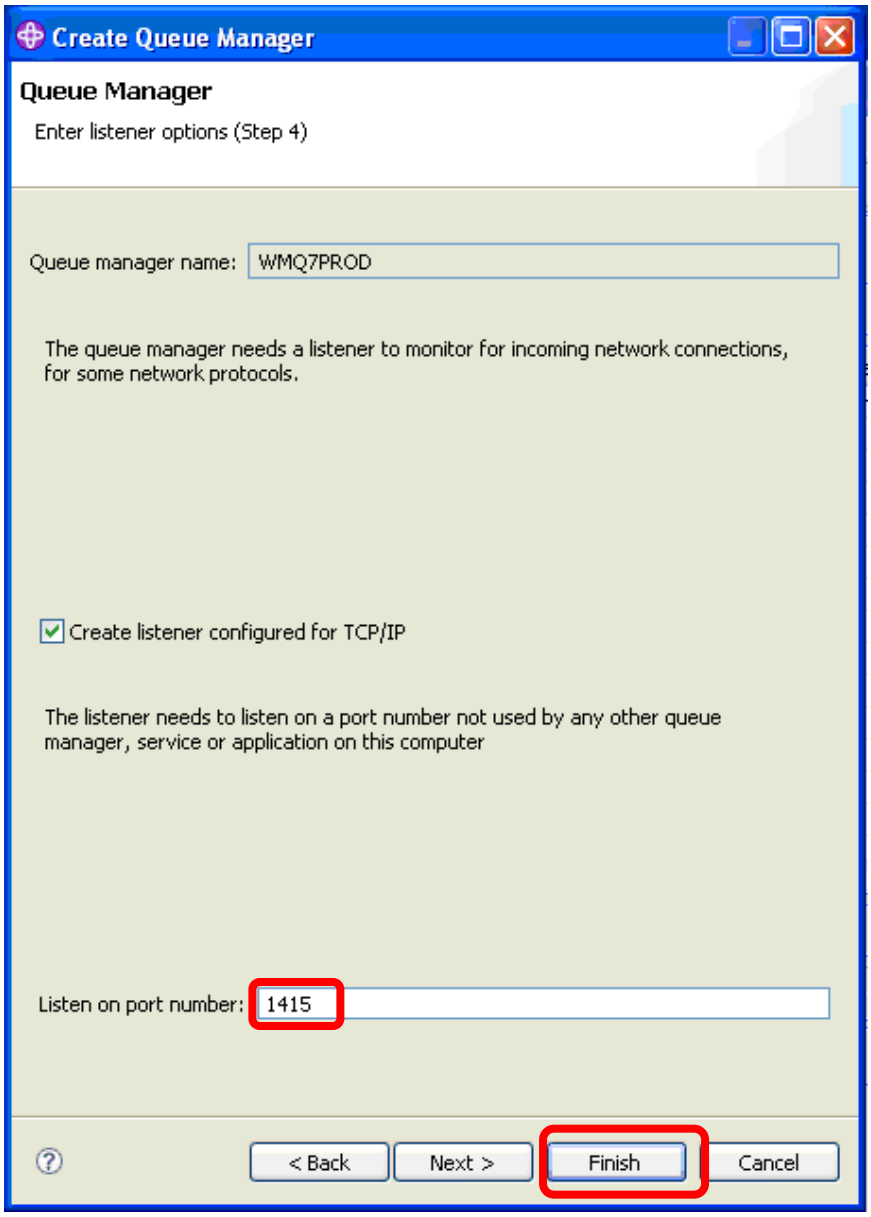
__3. Click the **Next** button to accept these defaults….



__4. Click the **Next** button to accept these defaults….



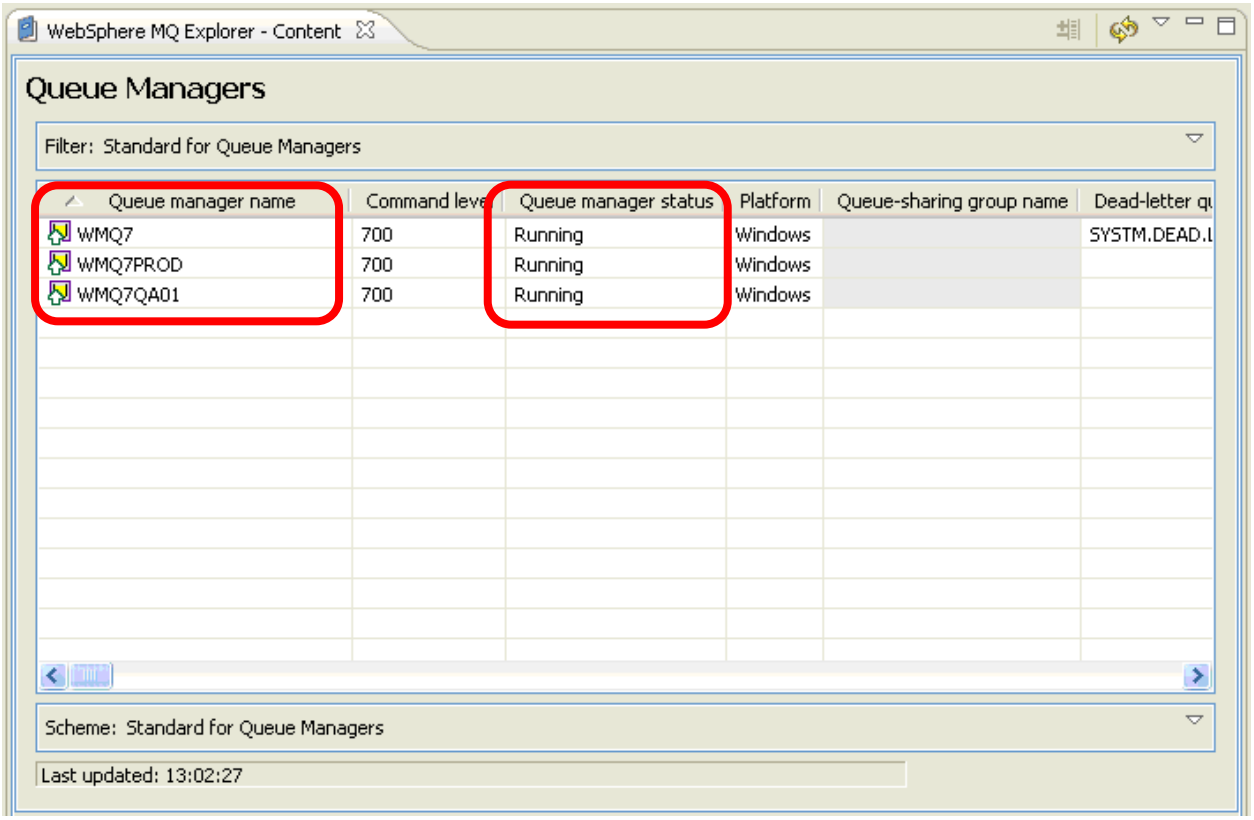
__5. Enter **1415** as the p**ort number** and click the **Finish** button.



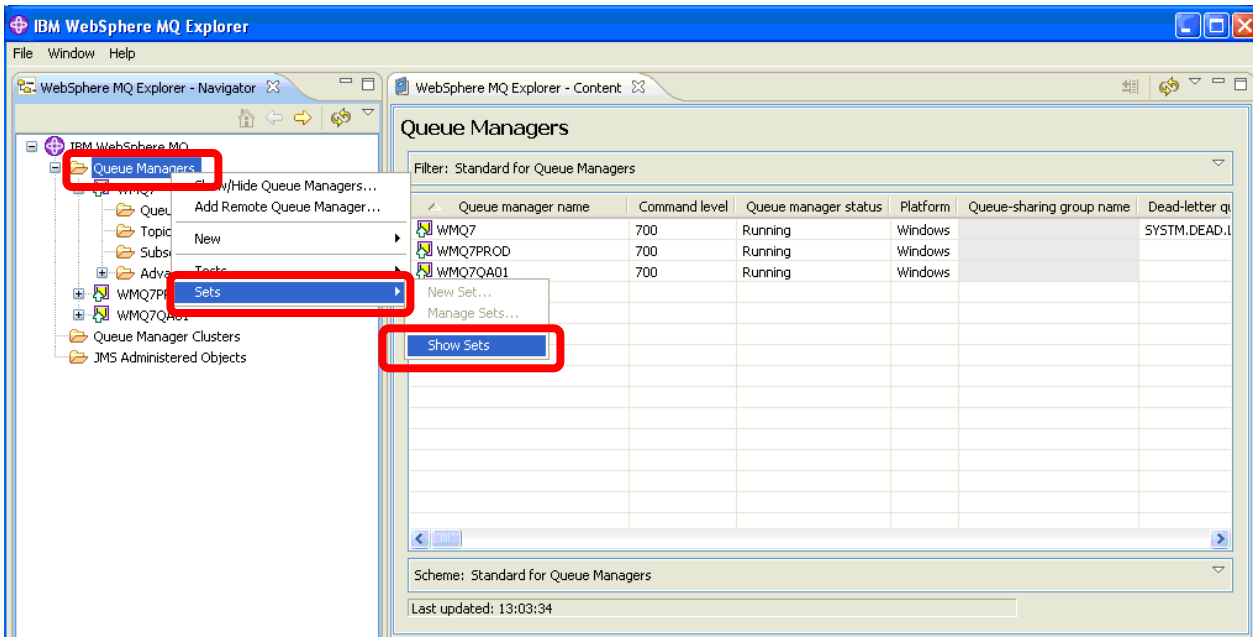
Using MQ Explorer and the steps shown above, create another queue manager and assign it the indicated port number for the listener:

WMQ7QA01 → port number = 1416

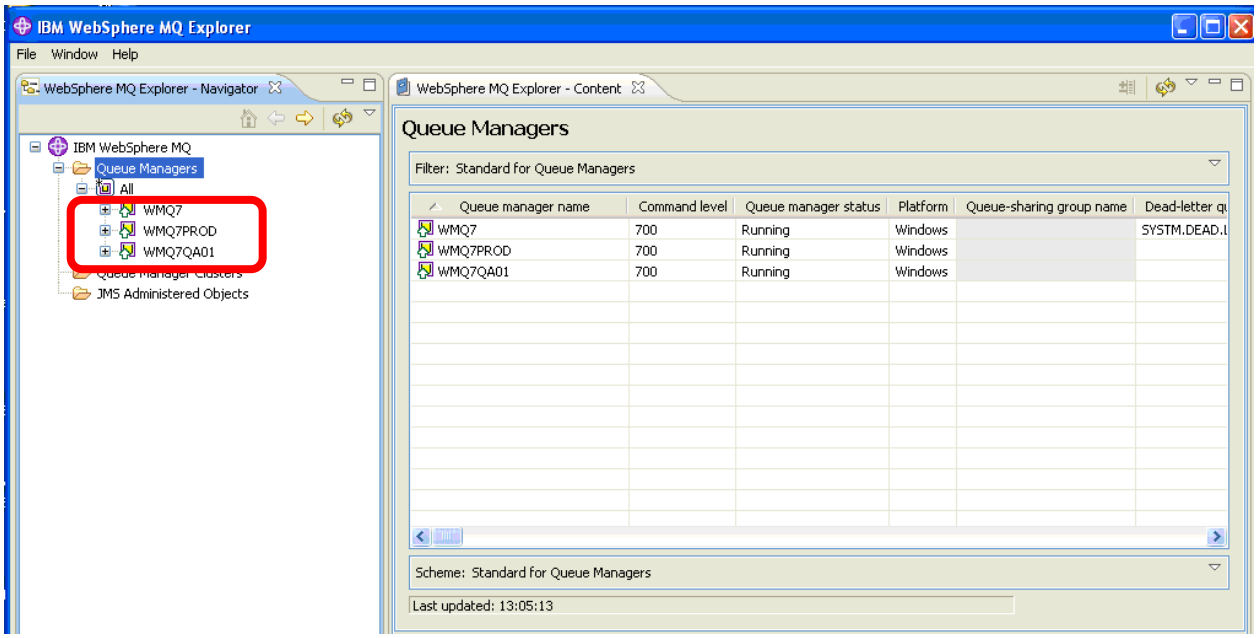
__6. Check the Content pane and make sure all three of the queue managers have been created, and that the **Queue manager status** for each is ***Running***.



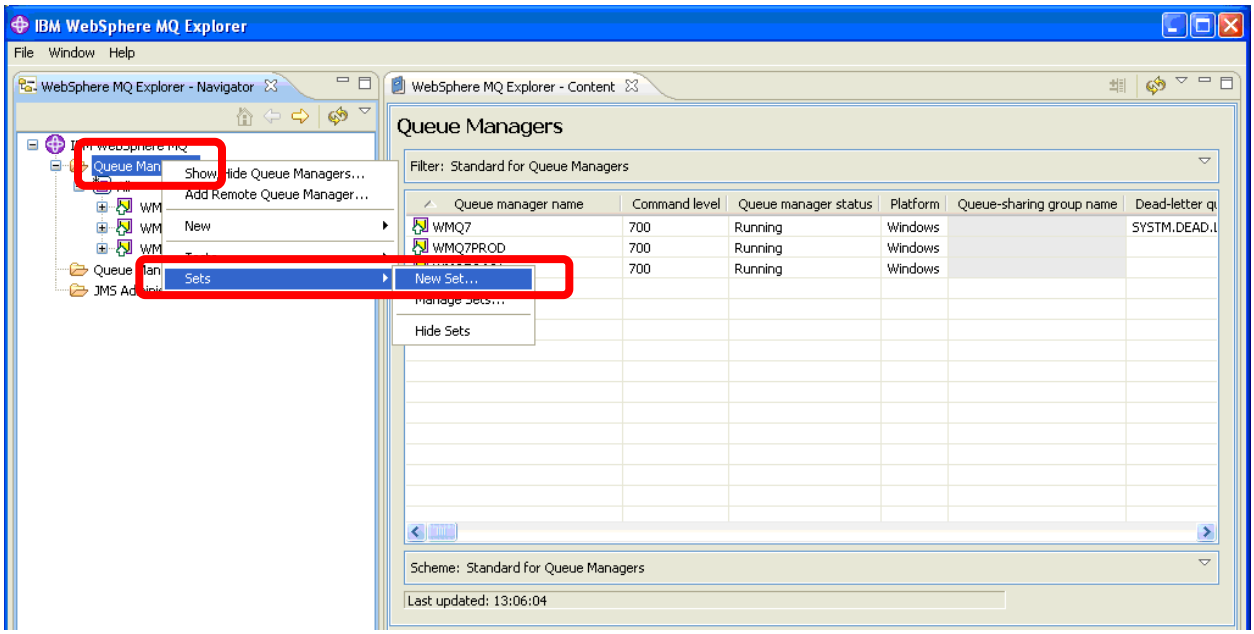
__7. To include Sets in the MQ Explorer display right-click on **Queue Managers** and select **Sets** then **Show Sets**



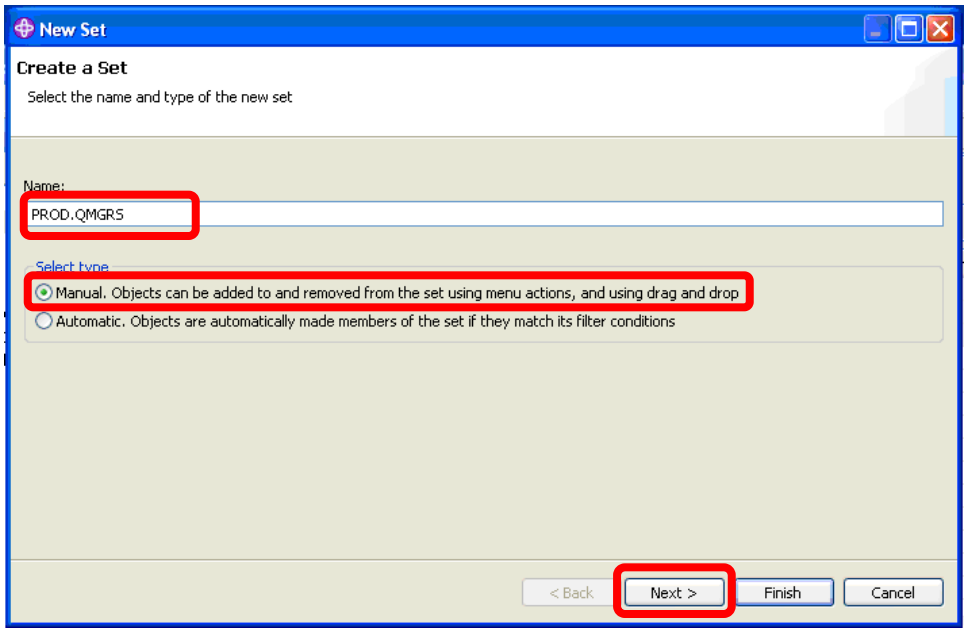
You now see that you have a **default set** called **All** that contains every queue manager that is defined.



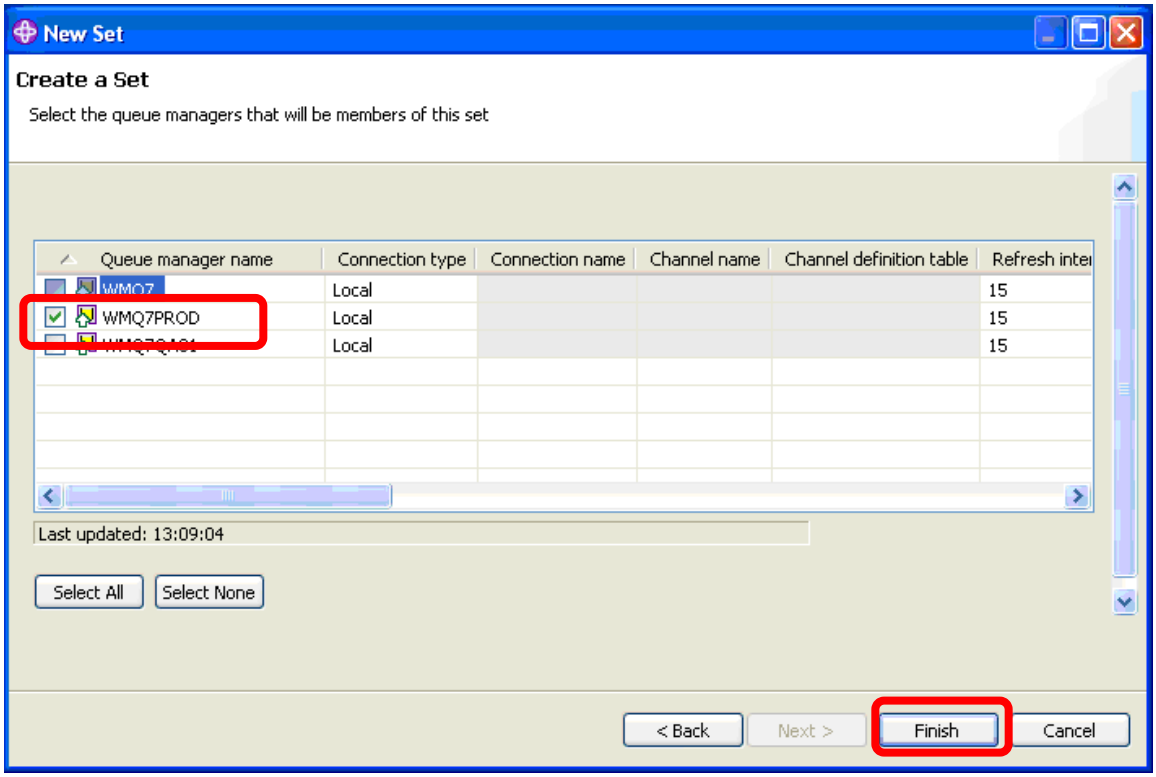
__8. To define a new Set, right-click on **Queue Managers** and select **Sets** then **New Set**.



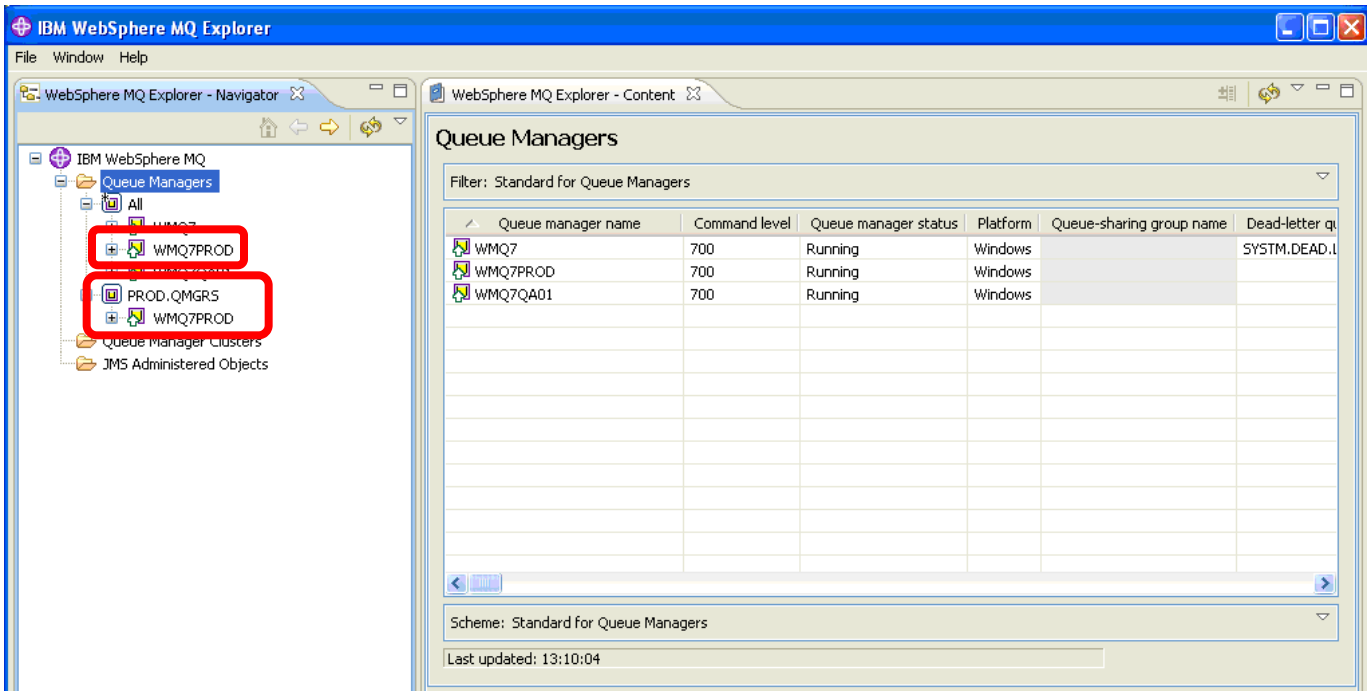
__9. Enter **PROD.QMGRS** as the name, leave the **Select type** as **Manual** and click on the **Next** button.



__10. Click the check box for **WMQ7PROD** to indicate that this queue manager is to be included in the Set and click the **Finish** button



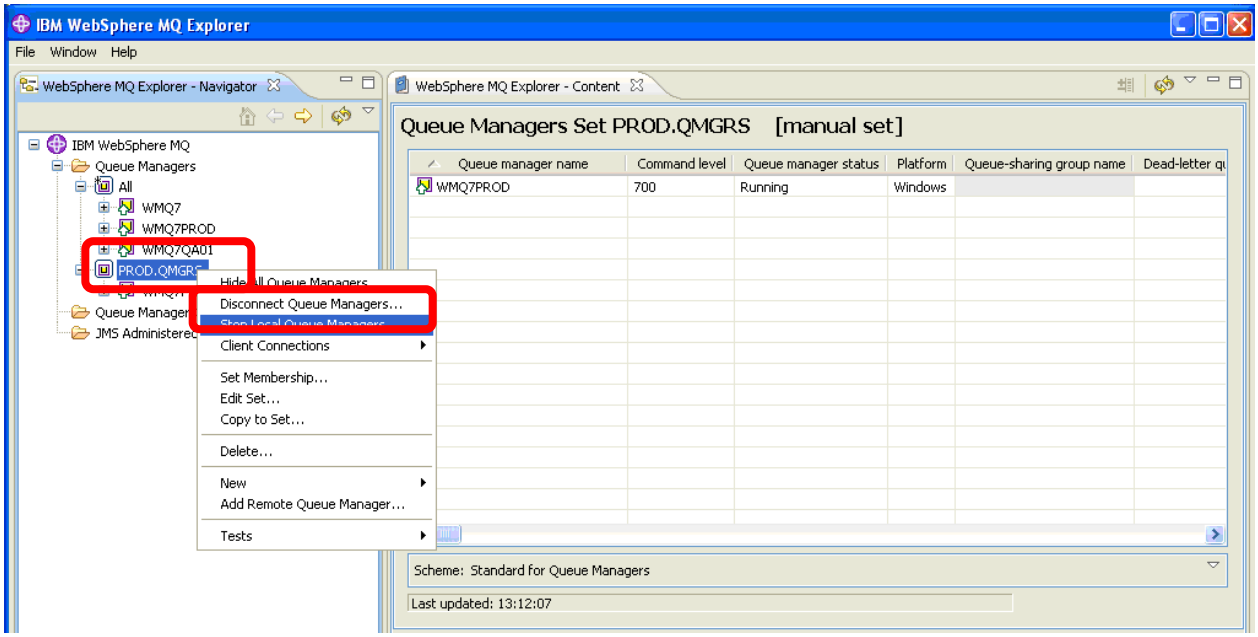
__11. Click on the plus sign to **expand** the **PROD.QMGRS** Set. Note that WMQ7PROD is a member of both the All Set and the PROD.QMGRS Set.



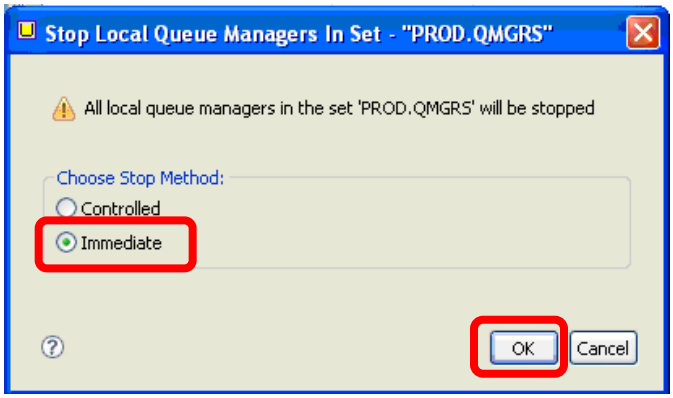
Once a Set has been created you can take an action against the entire Set of queue managers by issuing a single request. For example you can:

- Show/Hide All
- Connect/Disconnect All
- Start/Stop All Local
- Run Default/Custom Tests

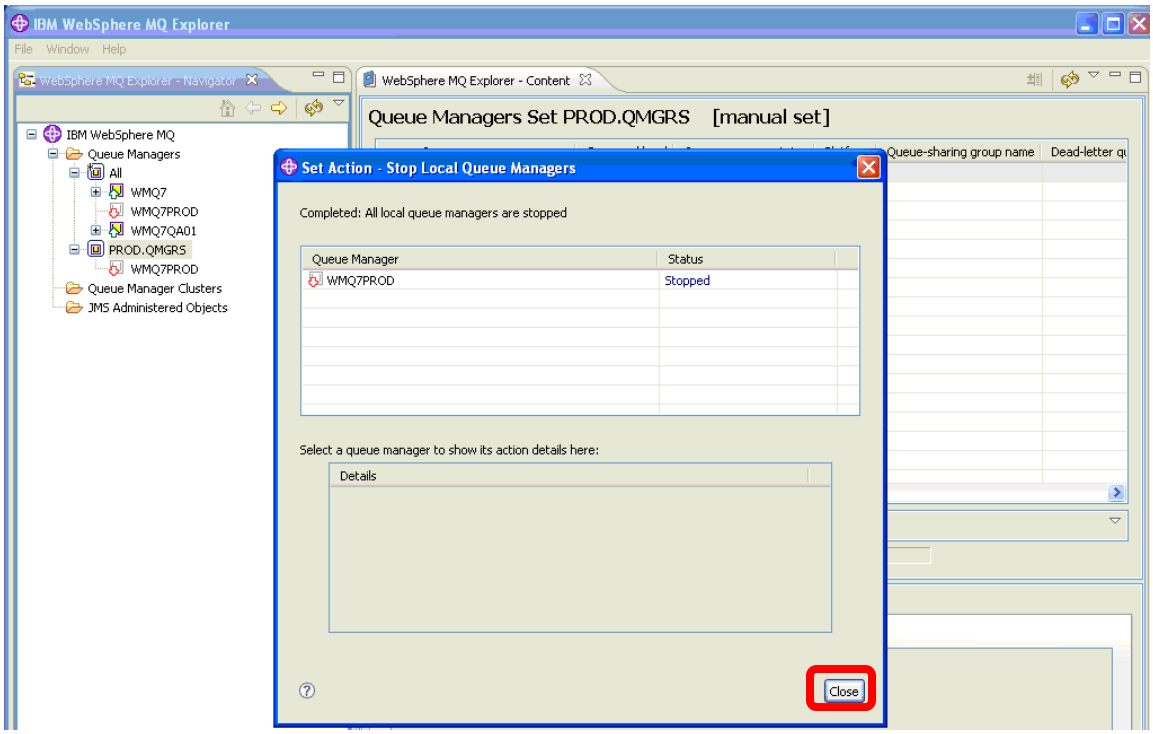
__12. To take some joint action against a Set of queue managers right-click on the Set and choose the desired action in the pulldown list. Right-click on **PROD.QMGRS** and select **Stop Local Queue Managers**…



__13. Choose **Immediate** for the **Stop Method** (not a good option for a production queue manager!) and click **OK**.

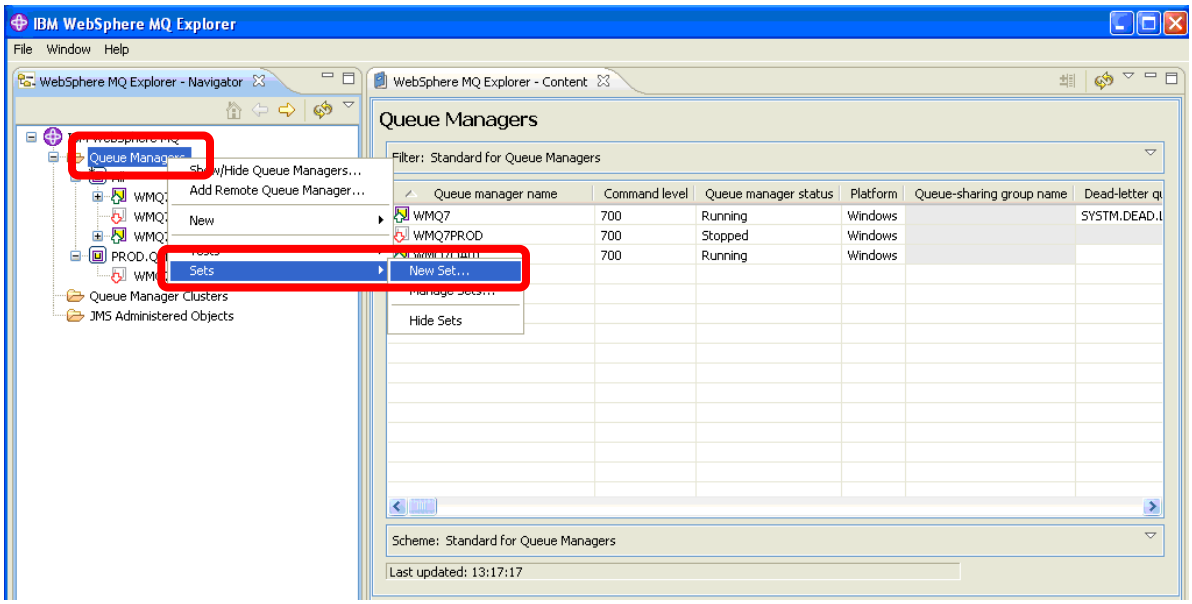


__14. After a few seconds the queue manager will show a status of **stopped**. Click the **Close** button to remove the Set Action panel.

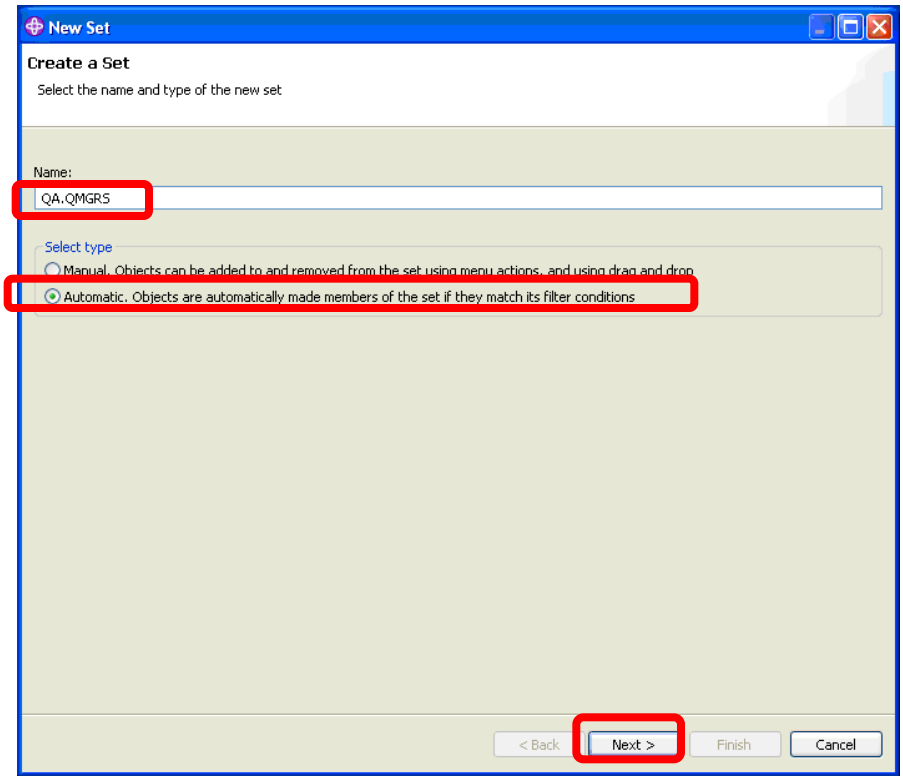


In the previous example, you performed your queue manager grouping in a manual fashion. Grouping can also be done automatically based on a filter you specify. To illustrate this you will create another Set and define it so that it will automatically add the appropriate queue manager(s) to the Set based on a portion of the queue manager name.

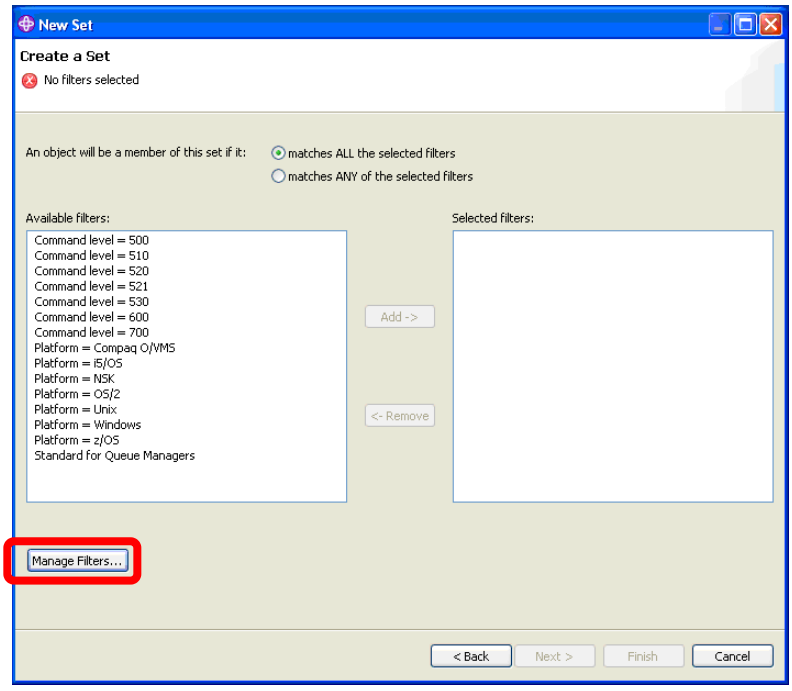
__15. Right-click on **Queue Managers** and select **Sets** then **New Set.**



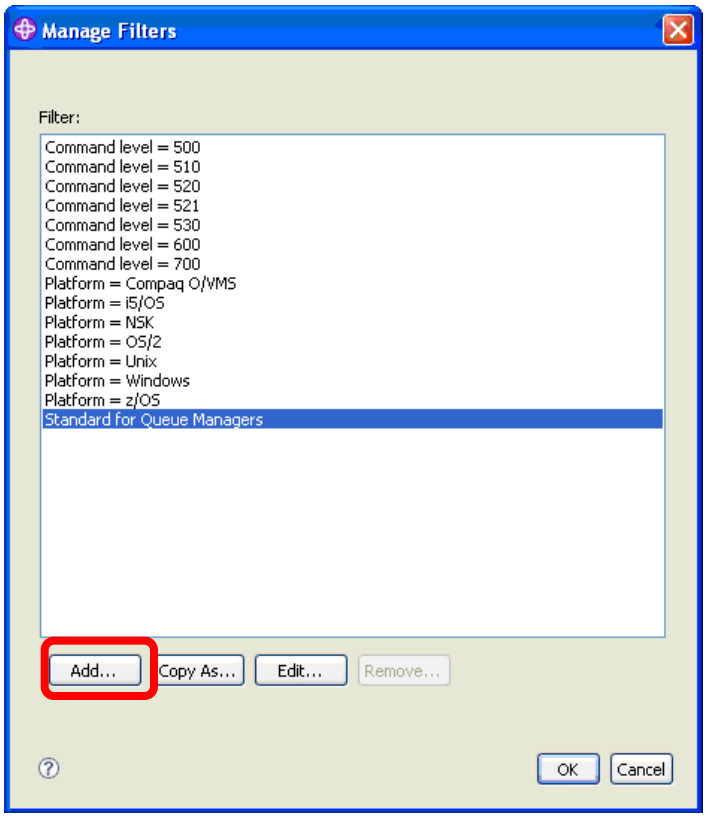
__16. Name the Set **QA.QMGRS**, select **Automatic** and click the **Next** button.



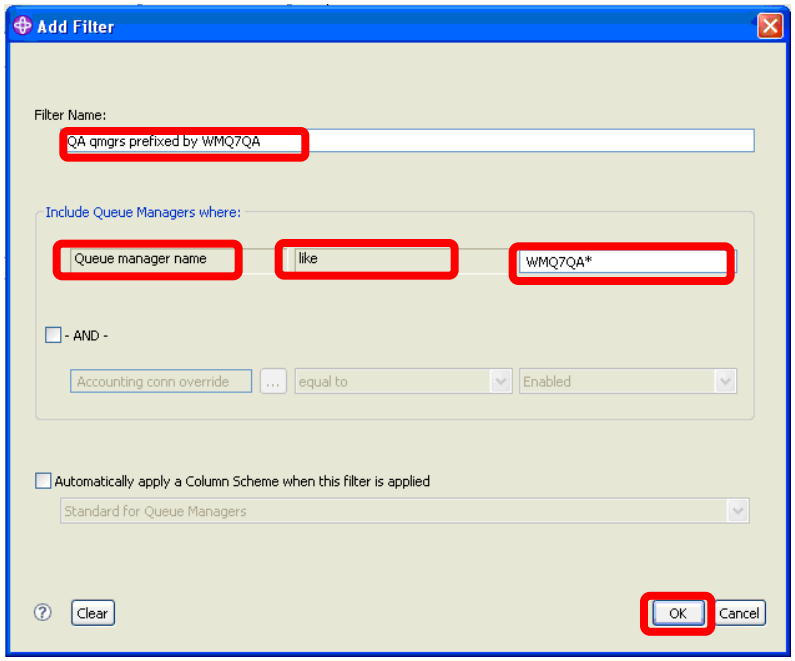
__17. This panel shows some predefined filters that can be used to choose queue managers for this Set. You can see that you could filter based on WebSphere MQ version and release level or on the operating system platform the queue manager is running on. You are going to create a new filter. Click on the **Manage Filters** button



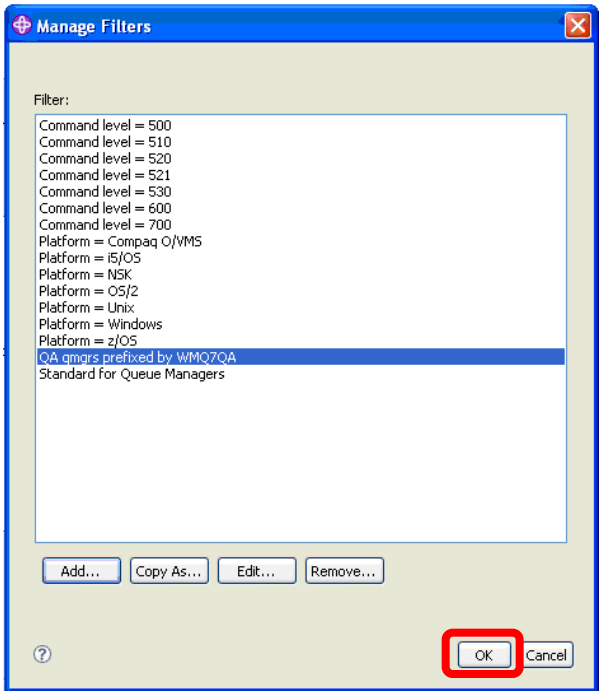
__18. Next, click the **Add** button to indicate that you want to add a new filter.



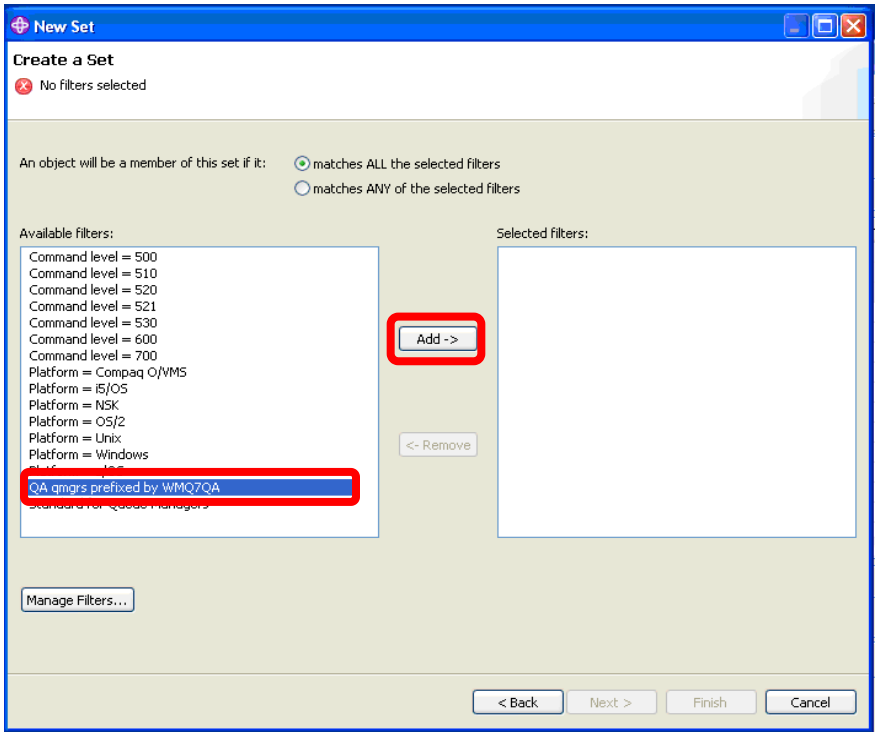
__19. Enter **QA qmgrs prefixed by WMQ7QA** for the **Filter Name**. Enter **WMQ7QA*** for the **Queue manager name like** field. The * is a wild card in this case….any queue manager name that begins with WMQ7QA will satisfy this filter. Click the **OK** button.



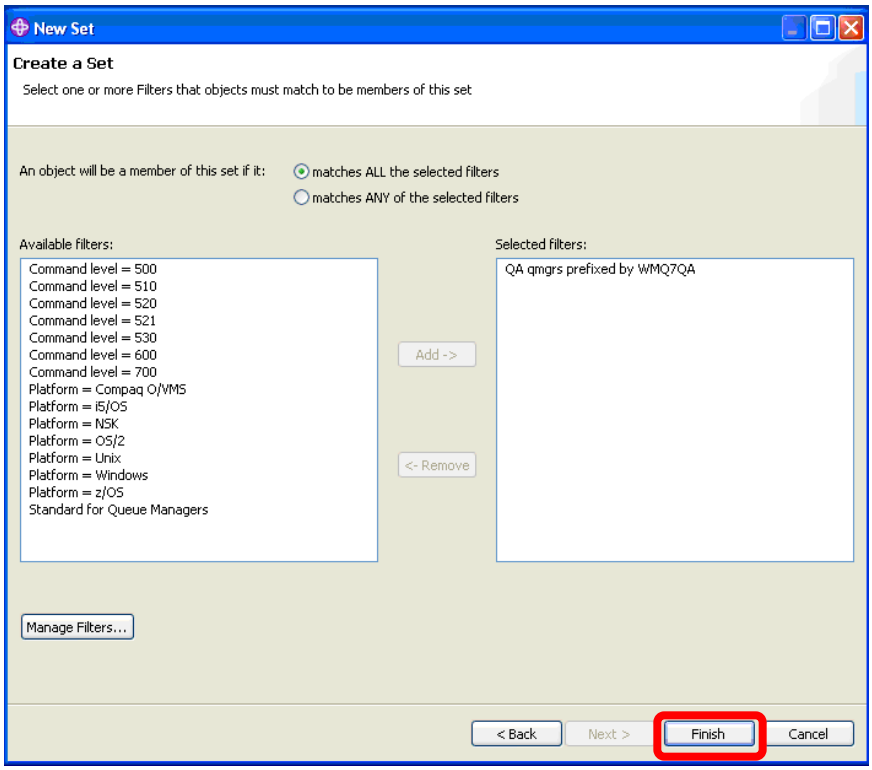
__20. The new filter now appears in the filter list. Click the **OK** button.



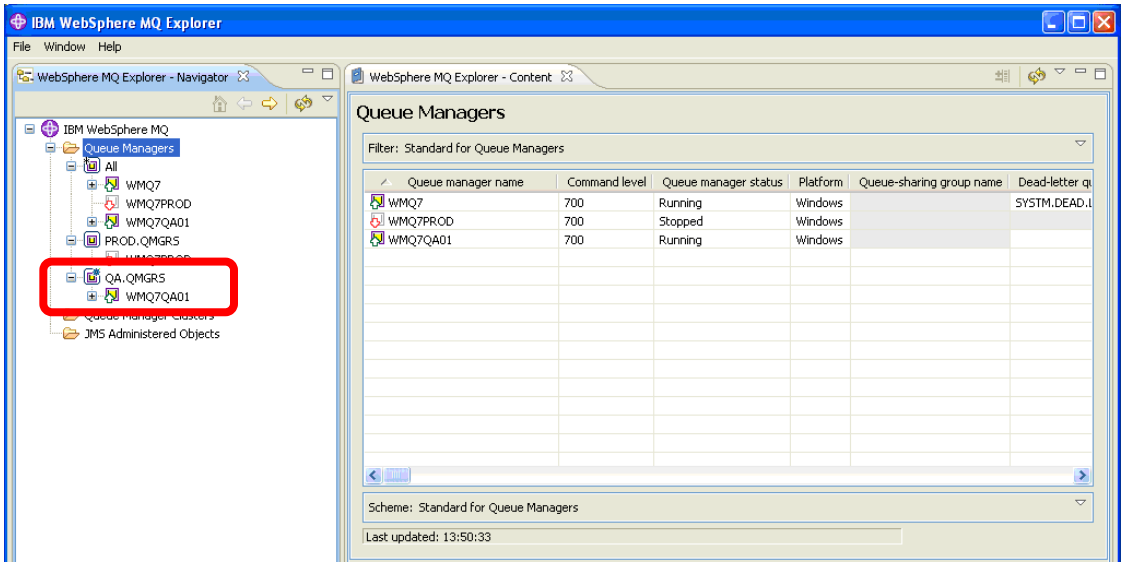
__21. **Click** the **QA qmgrs prefixed by WMQ7QA** in the **Available filters** list to select it then click the **Add** button to move it to the **Selected filters** box.



__22. Then click **Finish**.



__23. In the main MQ Explorer menu you can see that the new Set has been created and queue manager **WMQ7QA01** has been automatically added to the **QA.QMRGRS** group as it satisfied the filter you specified.

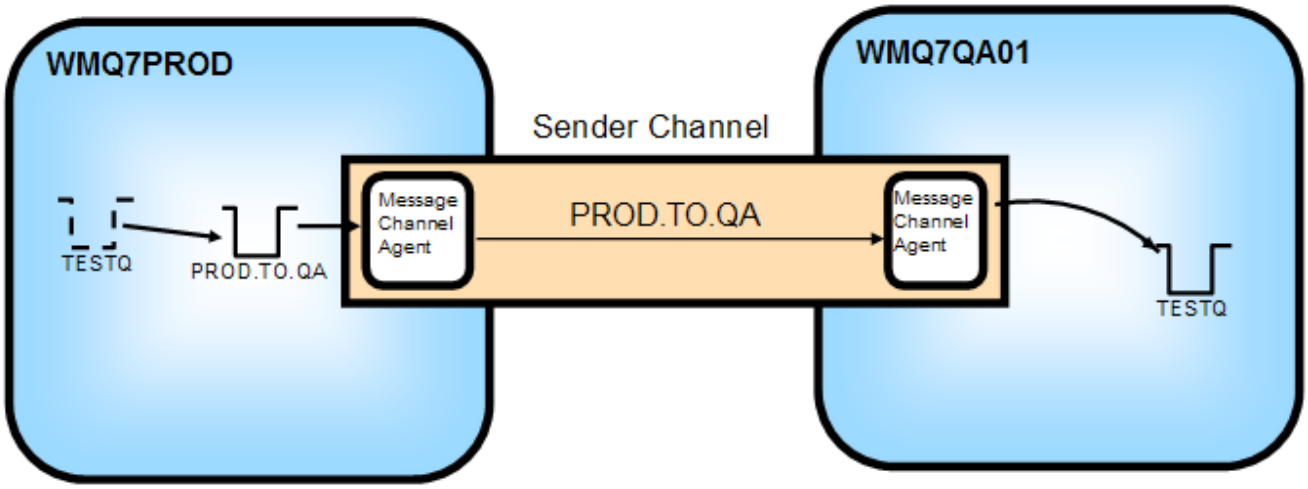


If you created a new queue manager whose name satisfied your filter it would also get automatically added to the QA.QMRGS Set.

This is the end of this portion of Lab 1

1.4 Distributed queuing

Your next step will be to explore distributed queuing. Since the primary reason to have an enterprise messaging product like WebSphere MQ is to move messages reliably between systems, this lab will give you a simple example of the steps involved in connecting two queue managers and moving messages between them. Here is a diagram of the configuration you will create:



Here are the MQ objects you will be creating and working with:

WMQ7PROD:

A *Remote queue* definition called **TESTQ**

A *Transmission queue* called **PROD.TO.QA**

A *Sender Channel* called **PROD.TO.QA**

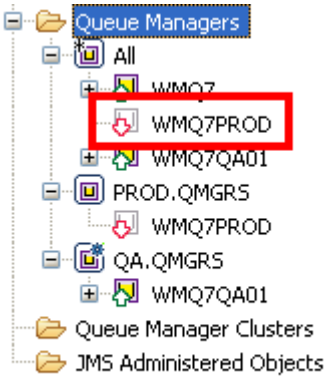
WMQ7QA01:

A *Local queue* called **TESTQ**

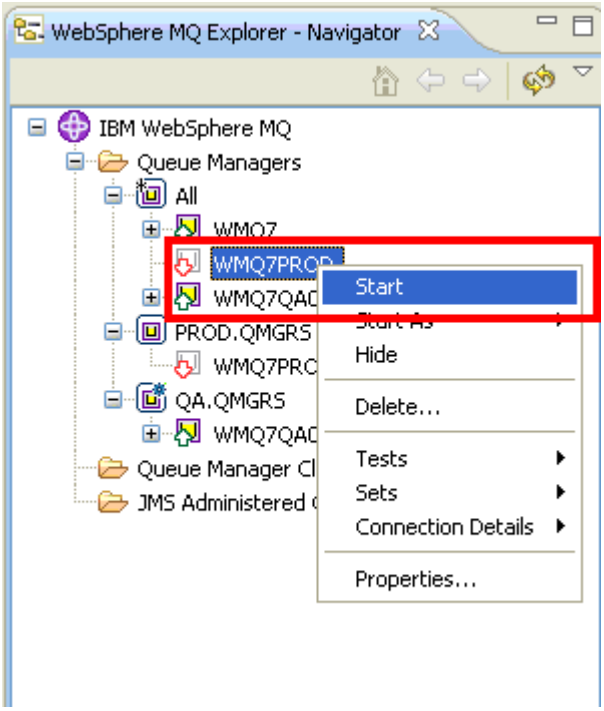
A *Receiver Channel* called **PROD.TO.QA**

You will then use one of the sample programs supplied with WebSphere MQ (called amqsput) to send messages from the WMQ7PROD queue manager, and another sample program called amqsget to read those messages off the destination queue on WMQ7QA01.

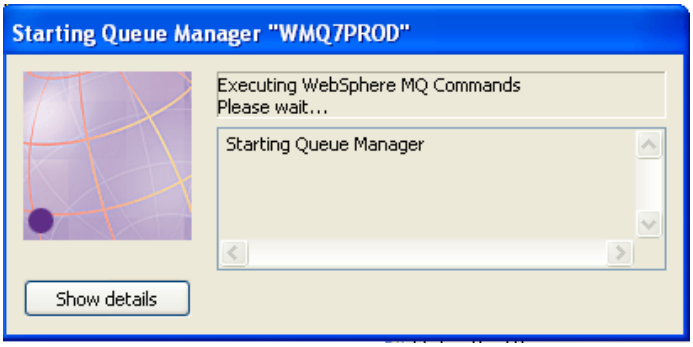
__1. In a previous step you stopped the WMQ7PROD queue manager. You want to restart it now. In the navigator pane, you should see the icon representing the WMQ7PROD queue manager with a **red** arrow pointing down, as in the picture below; this indicates that the queue manager is stopped (a **green** arrow pointing up indicates that the queue manager is running).



__2. Right-click on the WMQ7PROD queue manager, and select **Start**, as shown below.

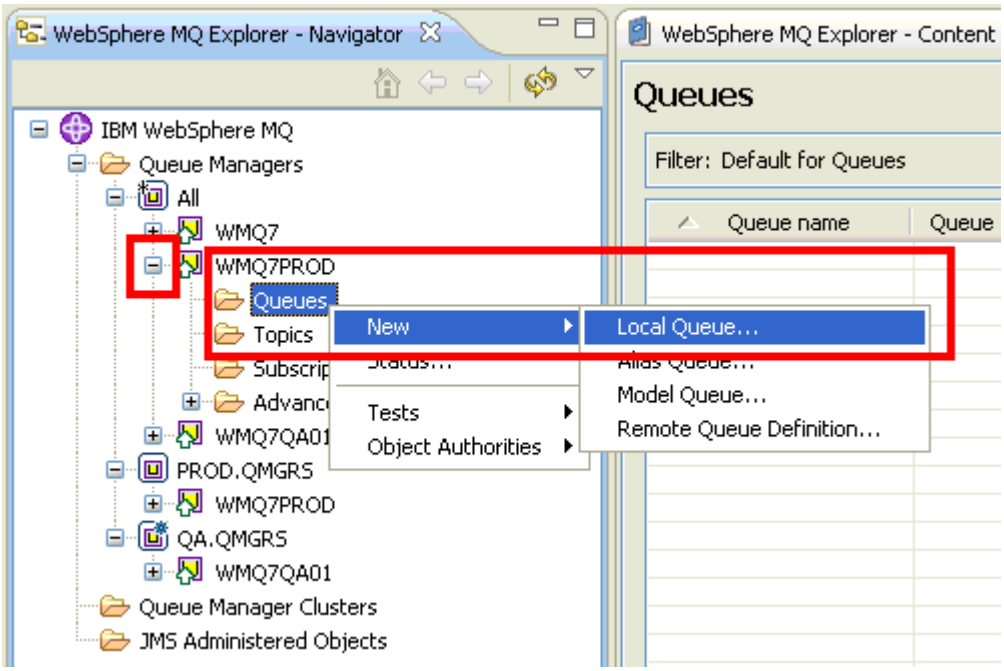


You will see a pop-up like the one below for several seconds while the queue manager starts.



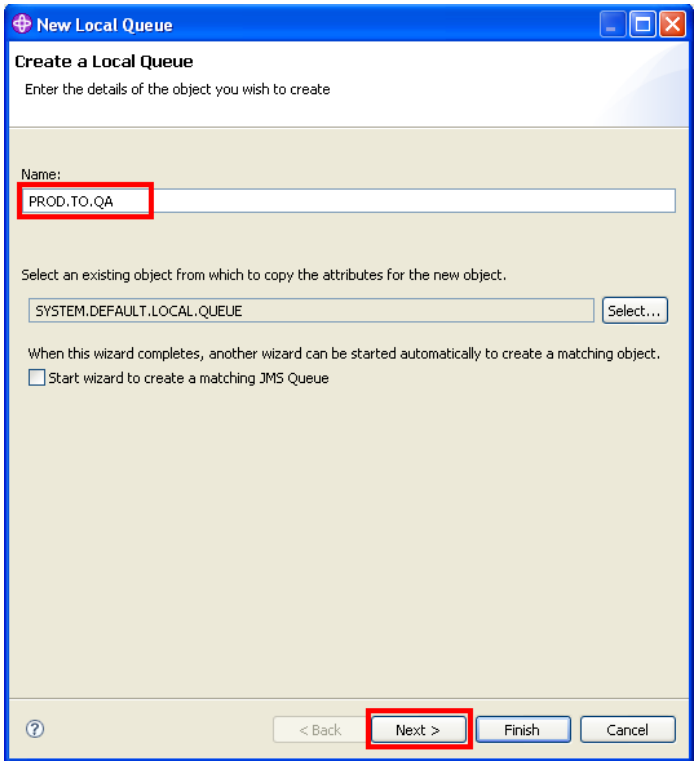
With WMQ7PROD running, you will now go through the steps needed to connect it to WMQ7QA01.

__3. First you will create a *Transmission queue*. Click on the plus sign next to WMQ7PROD to expand the tree, and then right-click on the **Queues** folder, select **New** and then **Local Queue**…

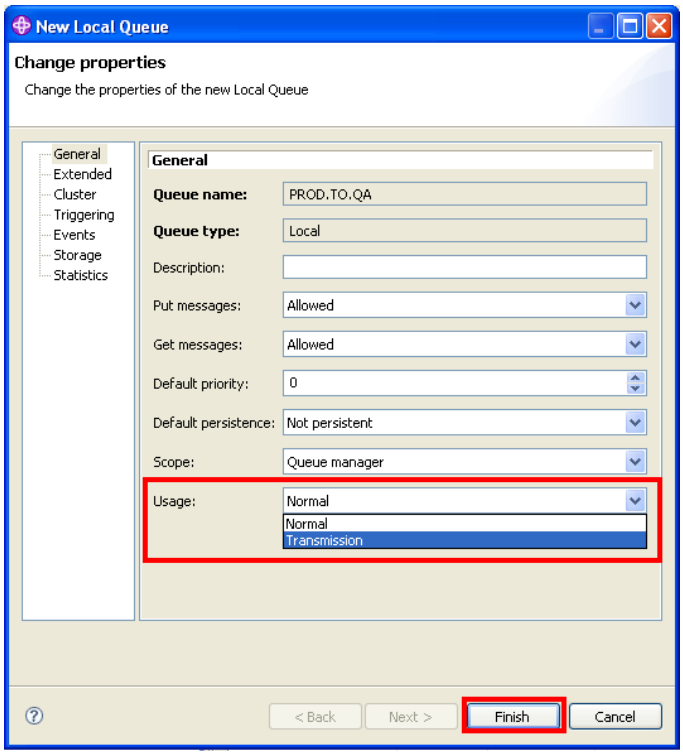


A Transmission queue is a special type of local queue, associated with a Sender Channel that holds messages safely until they are transferred to the queue manager on the other end of the channel. A typical nomenclature used for naming transmission queues and channels is *<sourceqm>.TO.<targetqm>*. You will use an abbreviated form for purposes of this lab, calling the transmission queue ***PROD.TO.QA***.

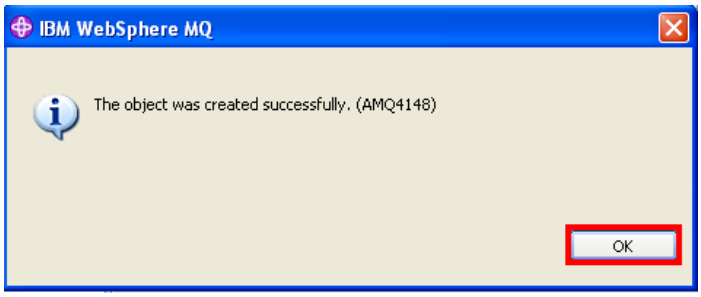
__4. Enter the name **PROD.TO.QA** as shown below, and click **Next**



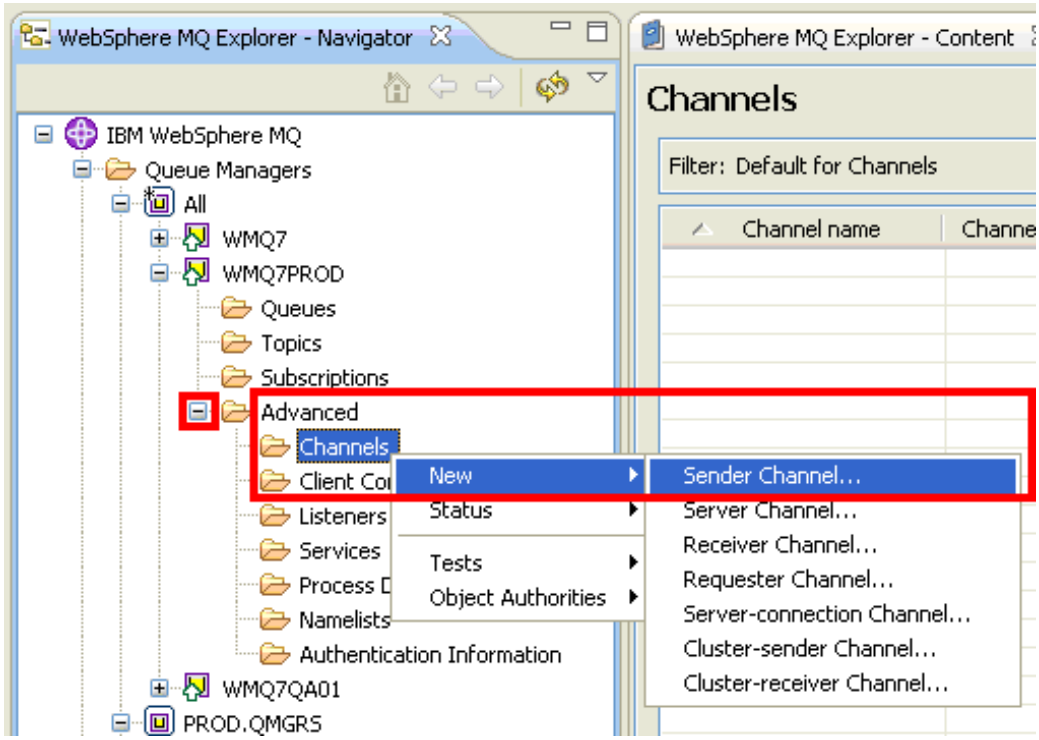
__5. Since this queue will be used as a transmission queue, indicate that by using the dropdown labeled **Usage**, and click **Finish**



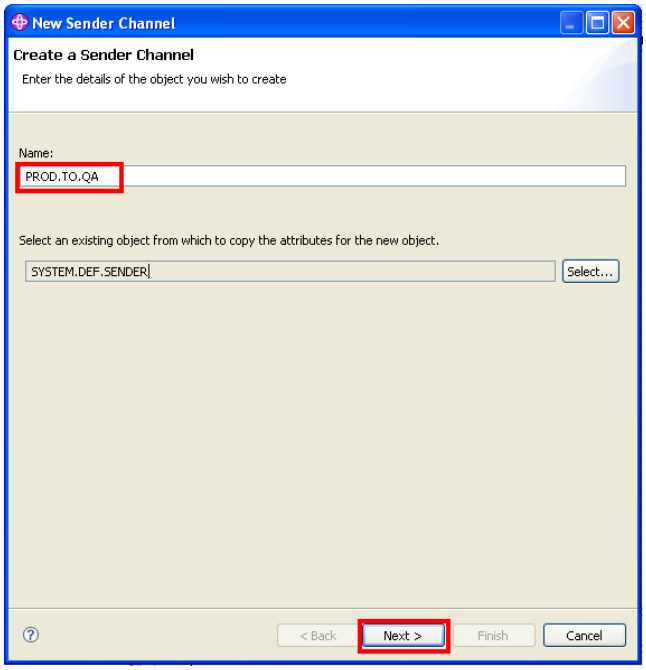
__6. Click OK on the confirmation popup



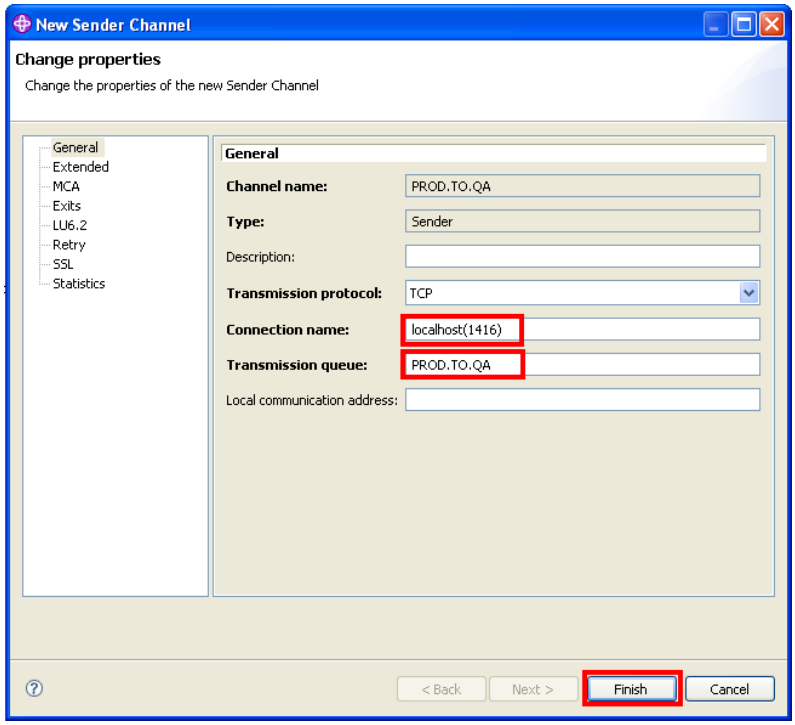
__7. Next you will create the Sender Channel this transmission queue will be associated with. Click on the plus sign next to the **Advanced** folder under WMQ7PROD to expand the tree, and then right-click on the **Channels** folder, select **New** and then **Sender Channel**



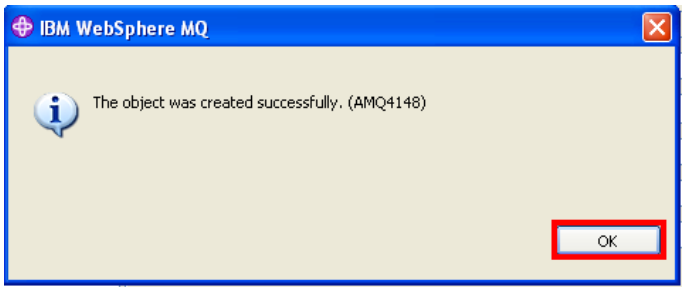
__8. Enter the name **PROD.TO.QA** as shown below, and click **Next**.



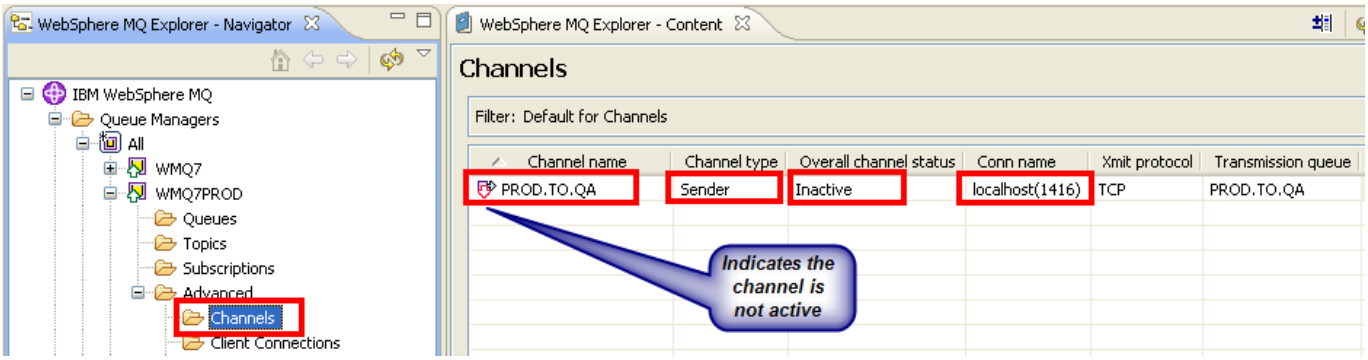
__9. You now need to identify the network location of the queue manager this channel is to connect to. Remember when you created the WMQ7QA01 queue manager, you assigned it a listener port of 1416. Specify that now, using **localhost** as the **Connection name**, as shown below. Also enter the name of the **Transmission queue** you created earlier, **PROD.TO.QA**. Then click **Finish**.



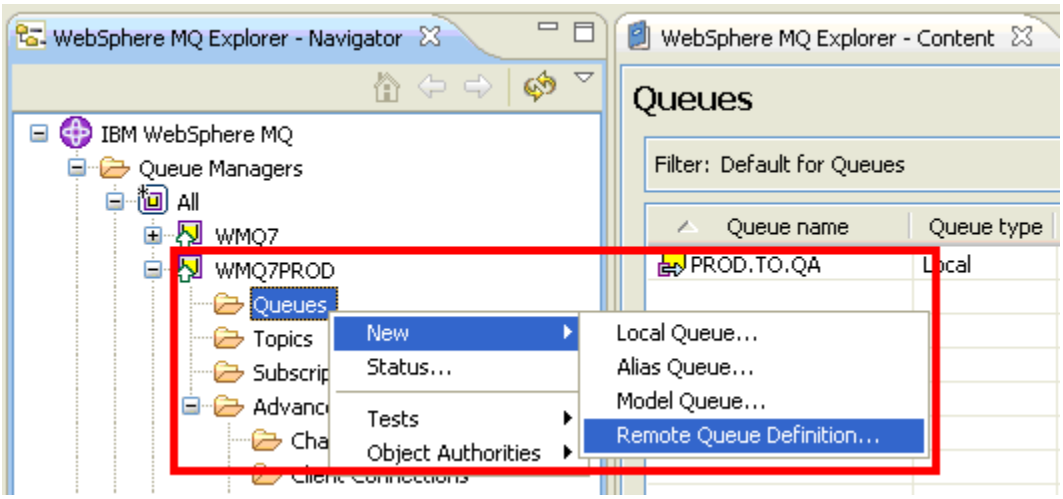
__10. Click OK on the confirmation popup. The channel has been created.



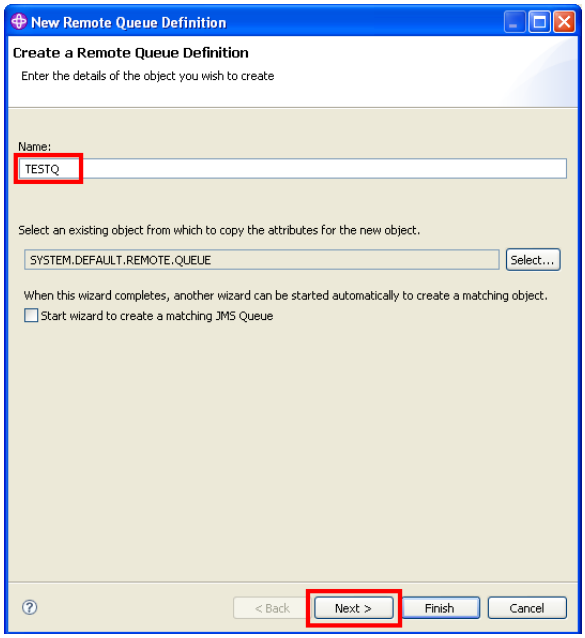
__11. In the Content pane on the right-hand side, you should see the details of the channel you just created. Notice that, in addition to the name, you see listed the **Channel type** of *Sender*, the **Overall channel status** as *Inactive*, and a **Conn name** of *localhost(1416)*



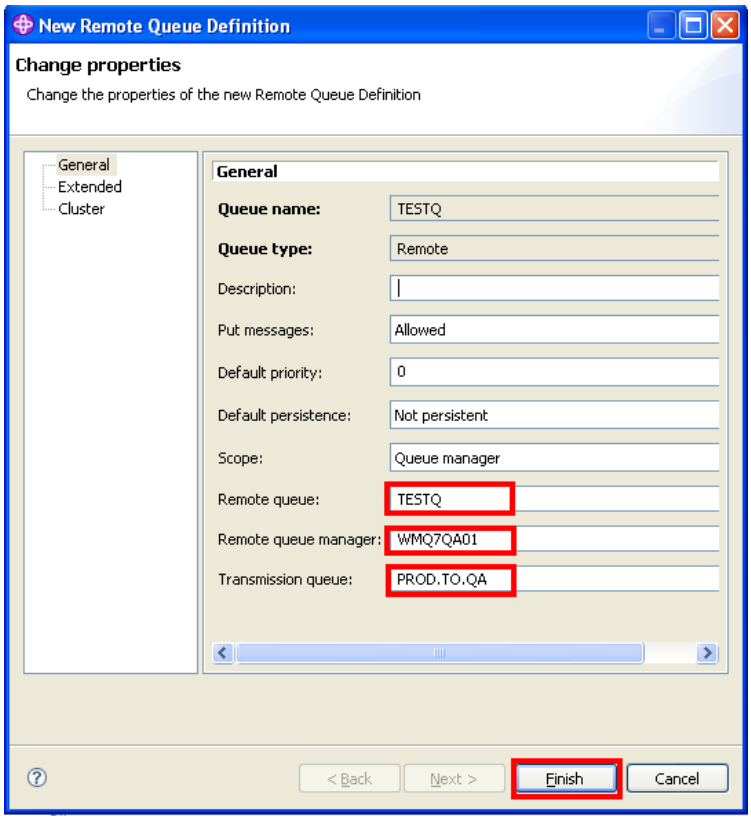
__12. The last task remaining on WMQ7PROD is to create a Remote queue definition. This definition will act as an alias of the real target queue that you will be creating shortly on WMQ7QA01. To create the remote queue definition, right-click on the **Queues** folder under WMQ7PROD, select **New** and then **Remote Queue Definition**…



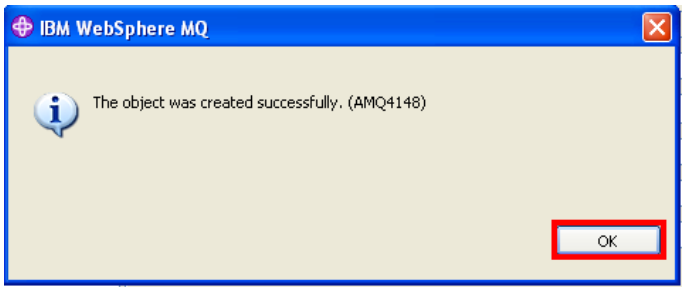
__13. Name the remote queue definition **TESTQ**, and click **Next**



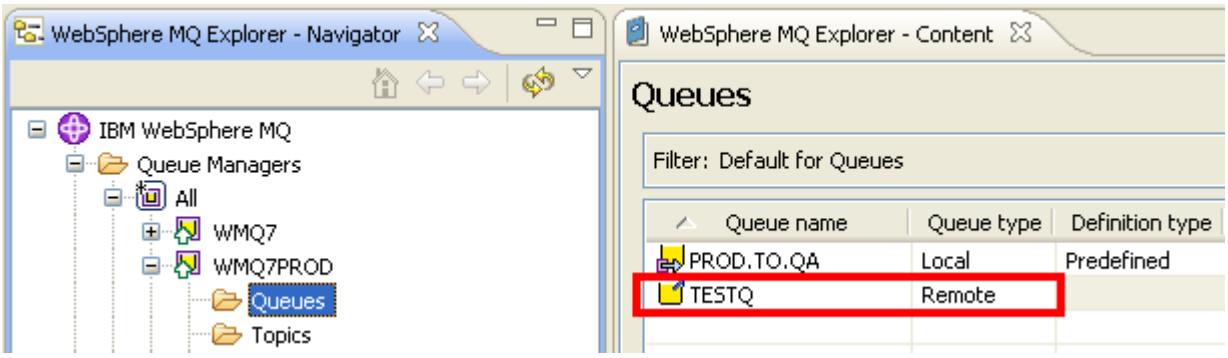
__14. Identify the name of the **Remote queue** (i.e. the name of the actual target queue) as ***TESTQ***. Specify the **Queue manager name** where the target queue resides (***WMQ7QA01***), and indicate the **Transmission queue** to be used to reach that queue manager (**PROD.TO.QA**). When those have been entered, click **Finish**.



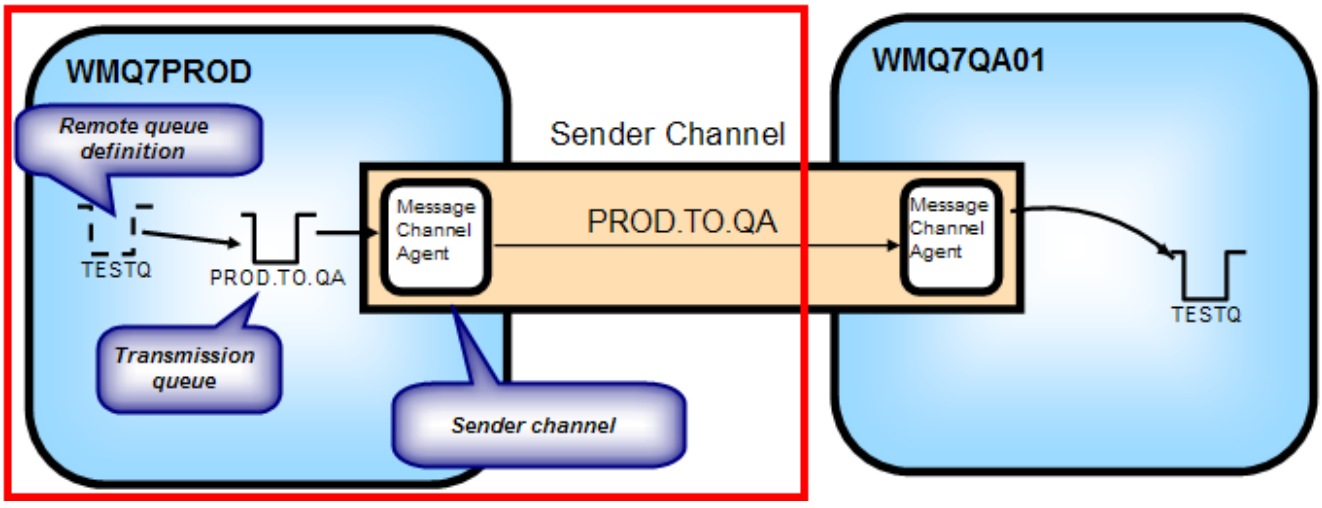
__15. Click OK on the confirmation popup. The remote queue definition has been created.



__16. You should now see your new queue definition under WMQ7PROD, with a **Queue type** of ***Remote***



Let's review what you have configured so far. Look at the left-hand side of the diagram below:

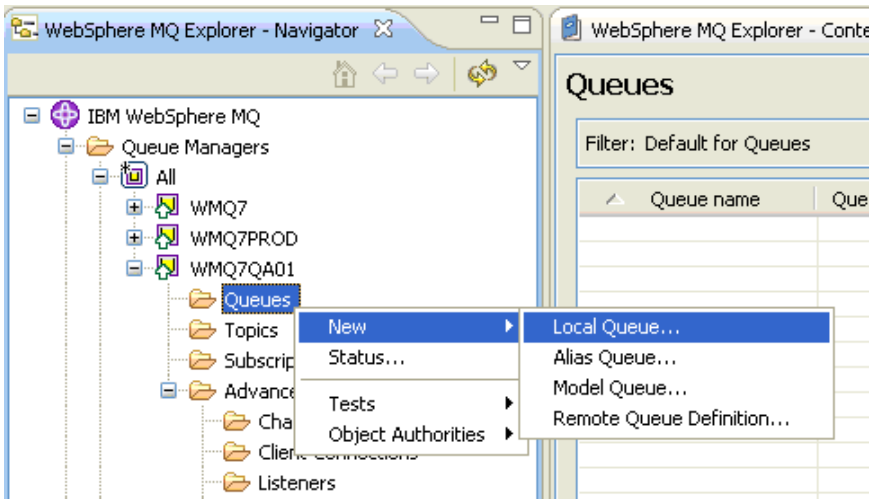


On WMQ7PROD you created a Remote queue definition of TESTQ, which points to a Transmission queue called PROD.TO.QA, which is associated with a Sender channel also called PROD.TO.QA that will connect to WMQ7QA01.

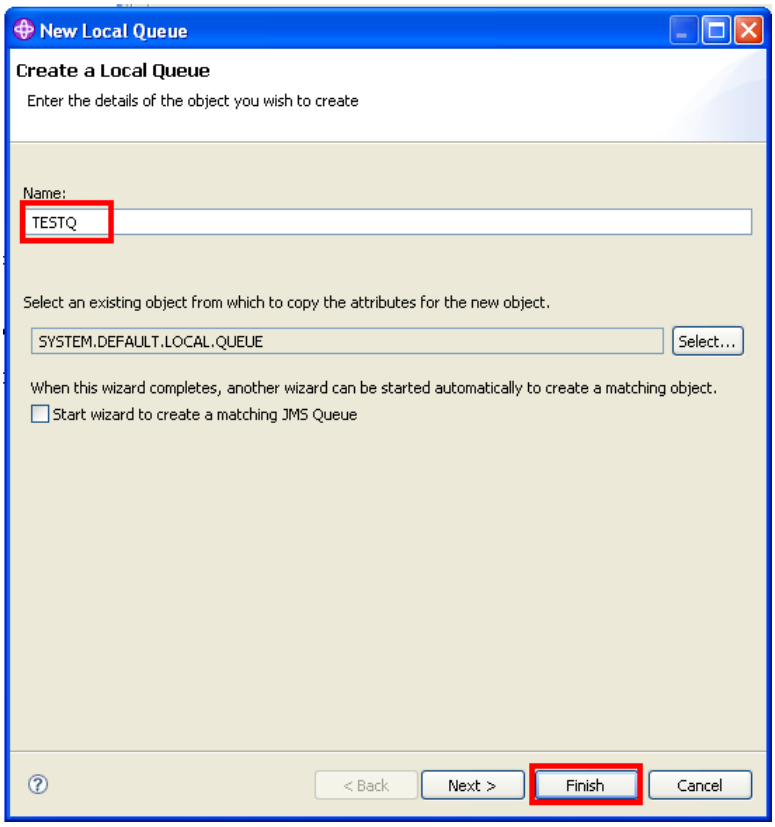
With the Sender side setup complete, you now need to create the corresponding definitions on WMQ7QA01.

__17. You previously created a Remote queue definition called TESTQ. As mentioned then, this definition is an alias for the actual local queue you are about to create.

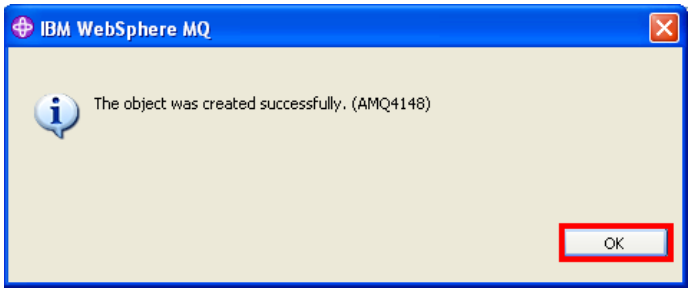
Expand the folder tree for queue manager WMQ7QA01, right-click on the **Queues** folder, click **New**, then **Local Queue**…



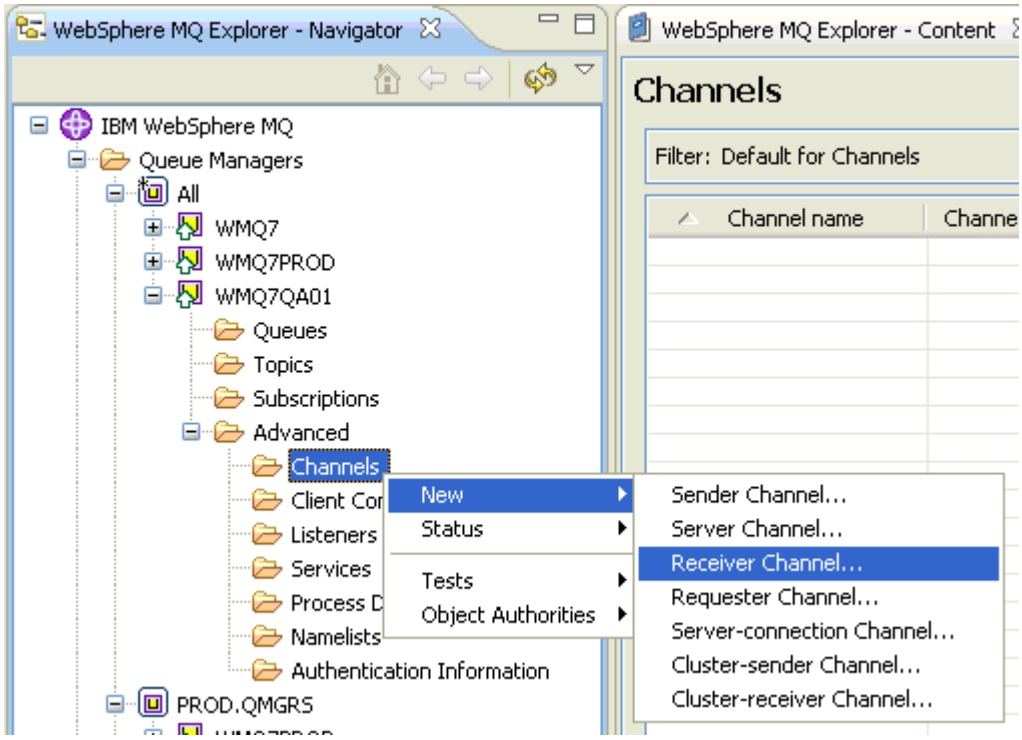
__18. Enter **TESTQ** as the Name. The default values are acceptable for this queue, so click **Finish**.



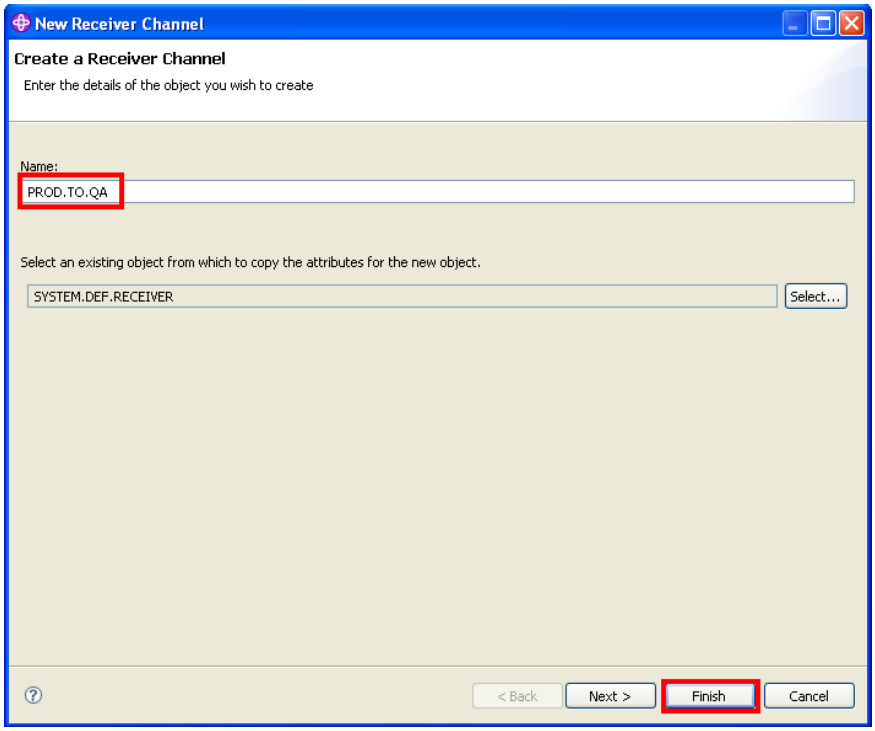
__19. Click OK on the confirmation popup.



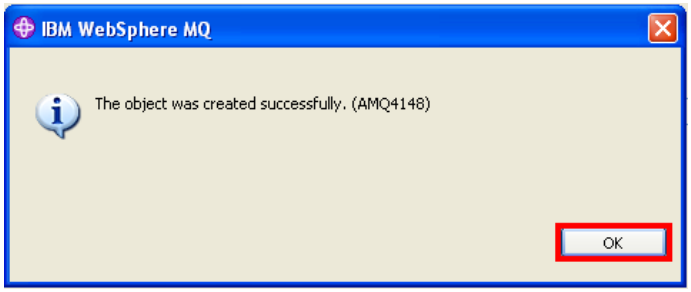
__20. Next you will create the Receiver Channel. This channel must have the same name as the corresponding Sender channel you created earlier. Expand the **Advanced** folder tree under WMQ7QA01, right-click on the **Channels** folder, then select **New**, then **Receiver Channel**…



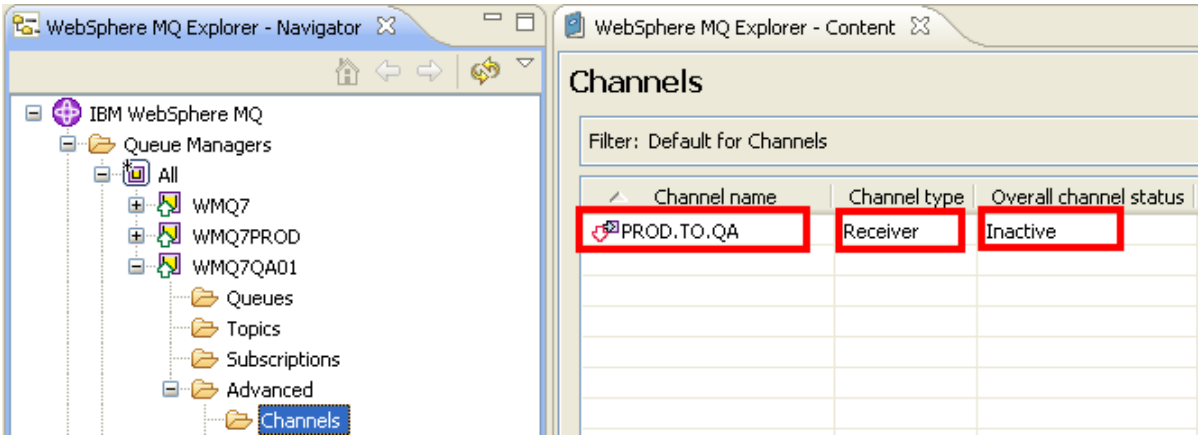
__21. Enter the name **PROD.TO.QA** as shown below. No other properties are required for a Receiver channel, so click **Finish**.



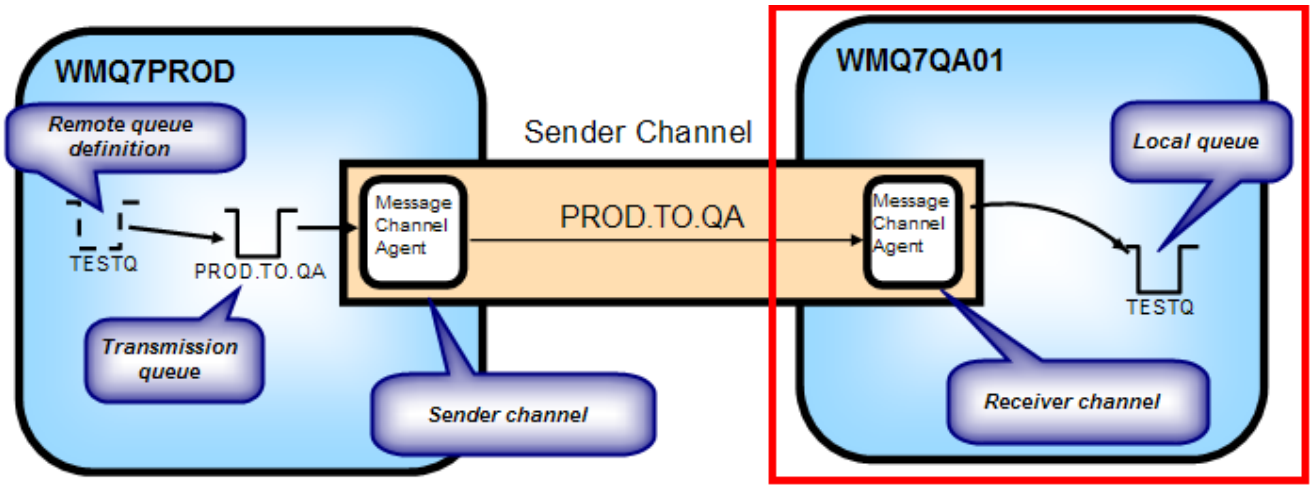
__22. Click OK on the confirmation popup. The channel has been created.



__23. In the Content pane on the right-hand side, you should see the details of the channel you just created. You should see the **Name**, ***PROD.TO.QA***, with a **Channel type** of ***Receiver*** and an **Overall channel status** of *Inactive*.



Your configuration should now be complete. Let's review what you have configured, looking at the right-hand side of the diagram below:



On WMQ7QA01 you created a Local queue called **TESTQ**, which will be the target queue for your test. You also created a **Receiver channel** as a partner for the Sender channel you created in WMQ7PROD, called **PROD.TO.QA**.

With the Receiver side setup complete, you are now ready to test your configuration and see how WebSphere MQ moves messages.

__24. Launch a command window by clicking the icon found on the taskbar

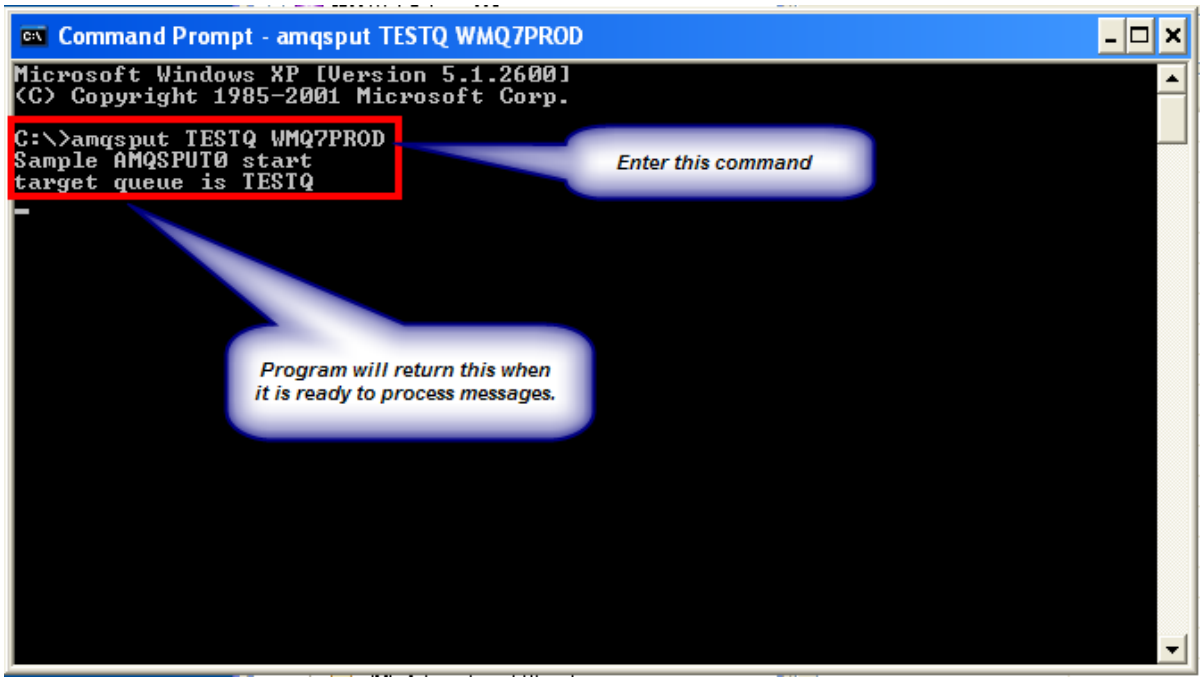
__25. For testing the putting (or sending) side, you will use a sample program supplied with WebSphere MQ called amqsput. The format of this command is:

amqsput <q_name> <qmgr_name>

For your test, you will be putting messages to the TESTQ remote queue definition on queue manager WMQ7PROD, so enter the following command:

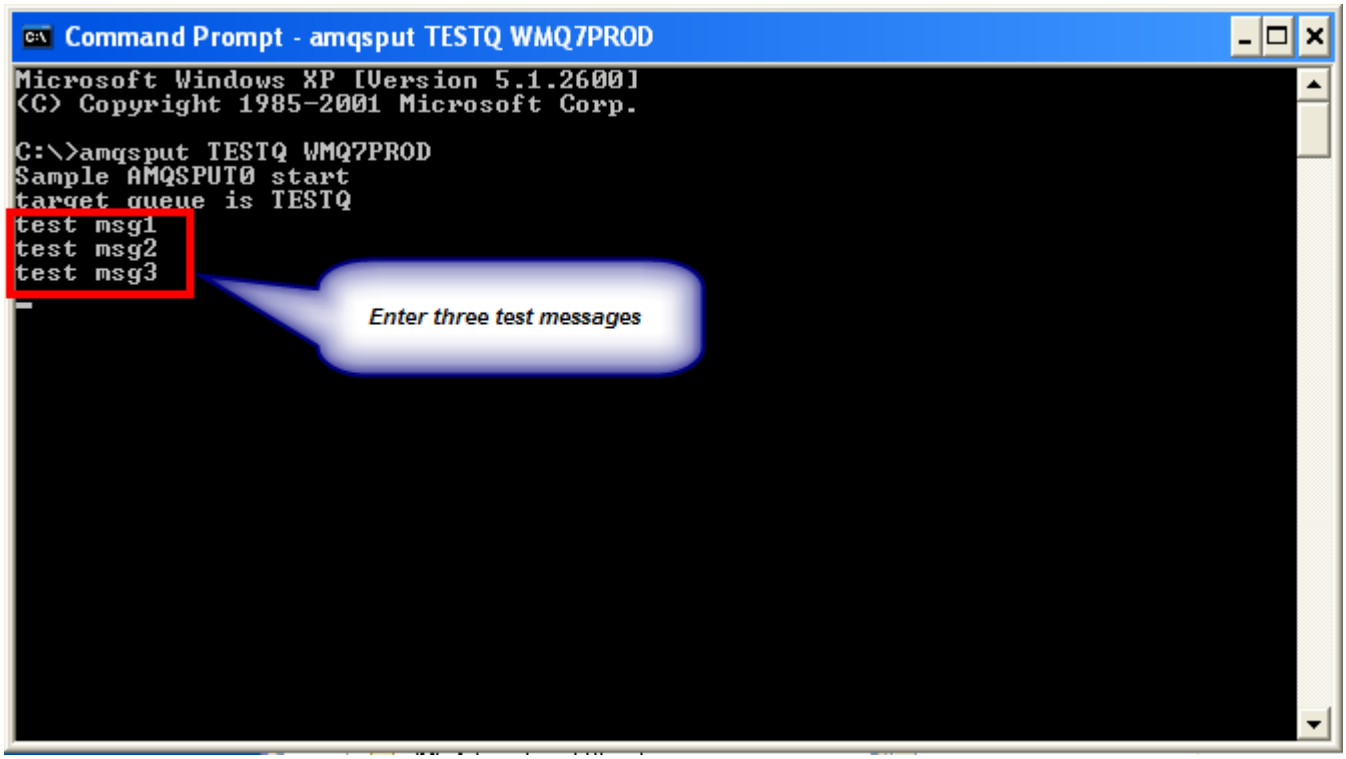
amqsput TESTQ WMQ7PROD

You should see the following when you enter this in the command window:



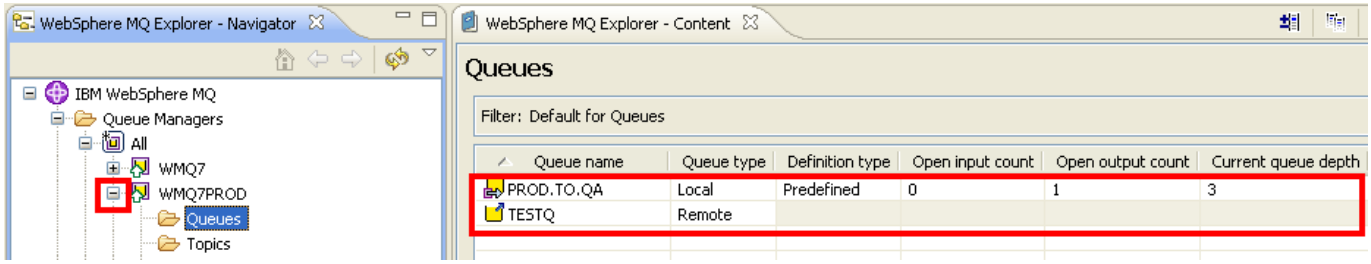
The amqsput sample program will take each line of text that you enter, and put each line as a separate message to the named queue.

__26. Enter three lines of text, as shown below:



Each line will be put on the target queue as a separate message.

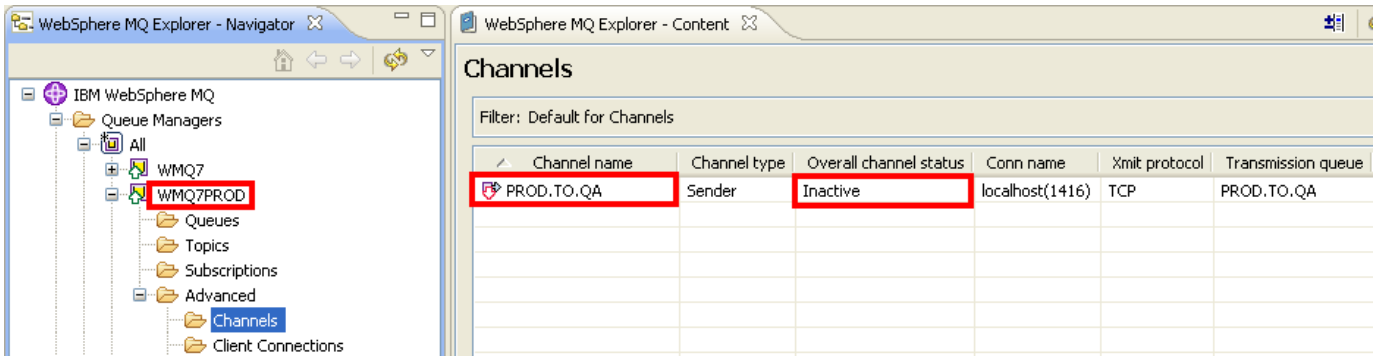
__27. Return to the MQ Explorer. In the Navigator pane, if necessary, expand the tree under WMQ7PROD by clicking the plus sign, and select the **Queues** folder. You should see the following:



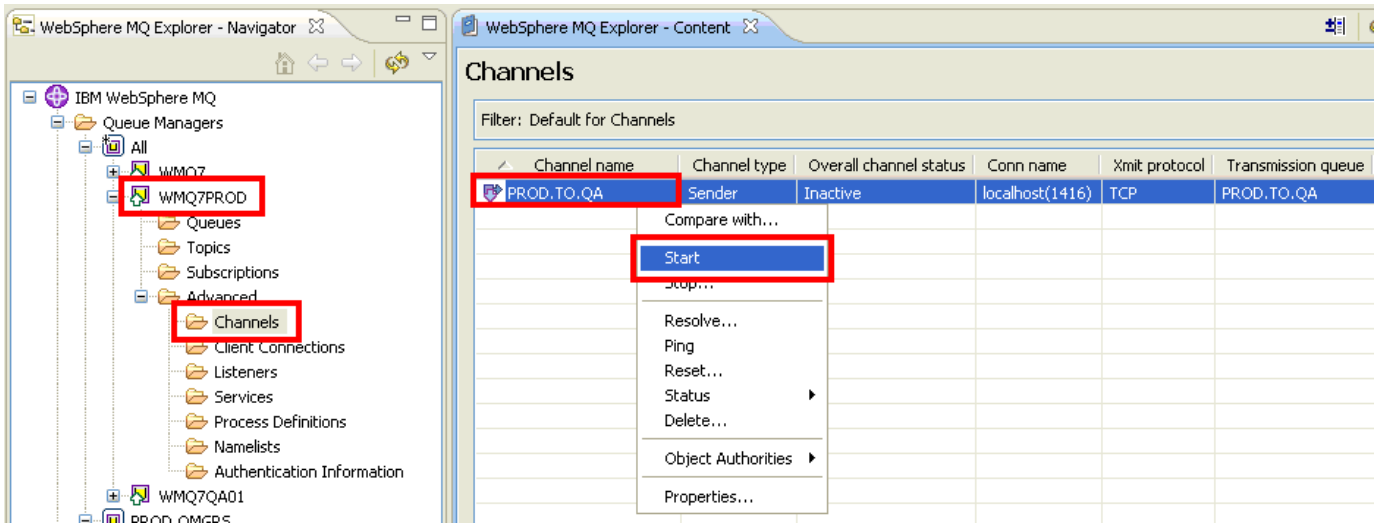
Look at the **Current queue depth** column and consider what you see here. Notice that TESTQ has no messages, and in fact this field is grayed out. Why???

Remember that a remote queue definition is like an alias; it is not an actual queue, but is instead a *reference* to one. In this case, it is a reference to the *transmission queue* associated with the channel that points to the queue manager where the actual TESTQ instance resides. If you look at the current queue depth of transmission queue **PROD.TO.QA** you see that it contains three messages. But why are they sitting here??? Why haven't they been transferred to the WMQ7QA01 queue manager?

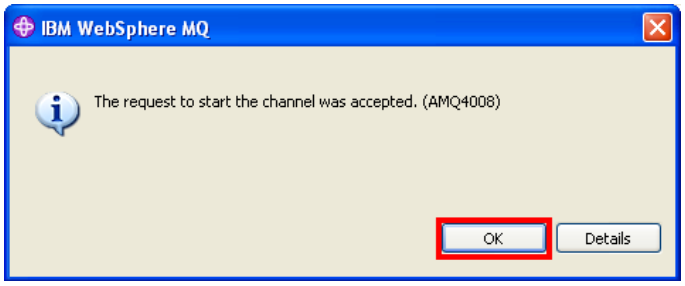
__28. To find out, click on the **Channels** folder on queue manager WMQ7PROD, and look at the **Overall channel status** for Channel **PROD.TO.QA**. Notice that the channel is **Inactive**! In order to move messages, the channel must have a status of Running. It is possible to configure the channel such that it will start *automatically* when a message arrives on its transmission queue, but we did not do that for purposes of this lab. Instead, you will start the channel manually.



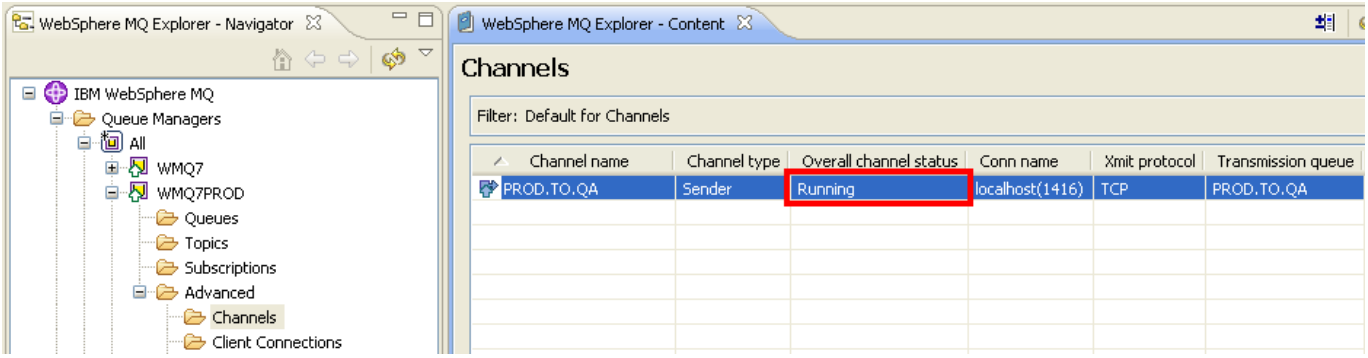
__29. Right-click on the **PROD.TO.QA** Sender channel entry, and click **Start**



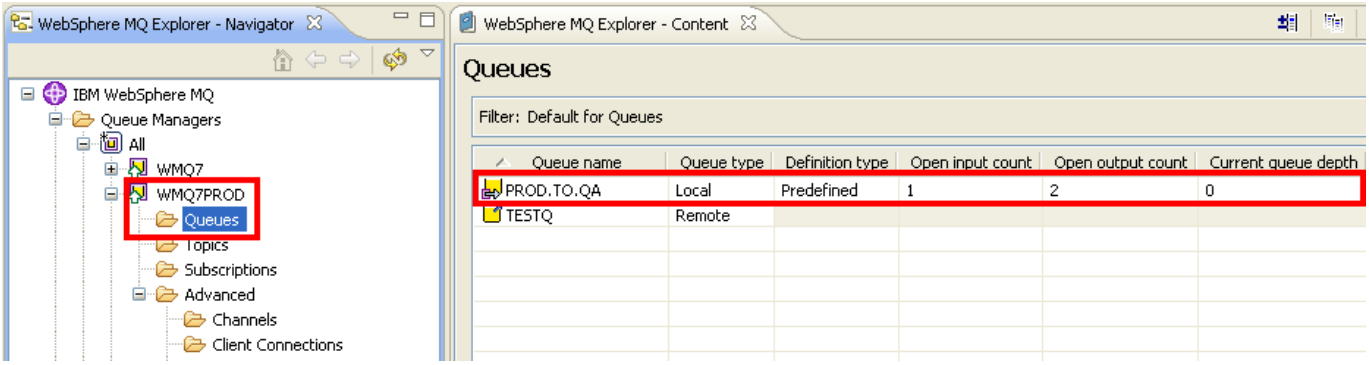
__30. Click **OK** on the confirmation popup.



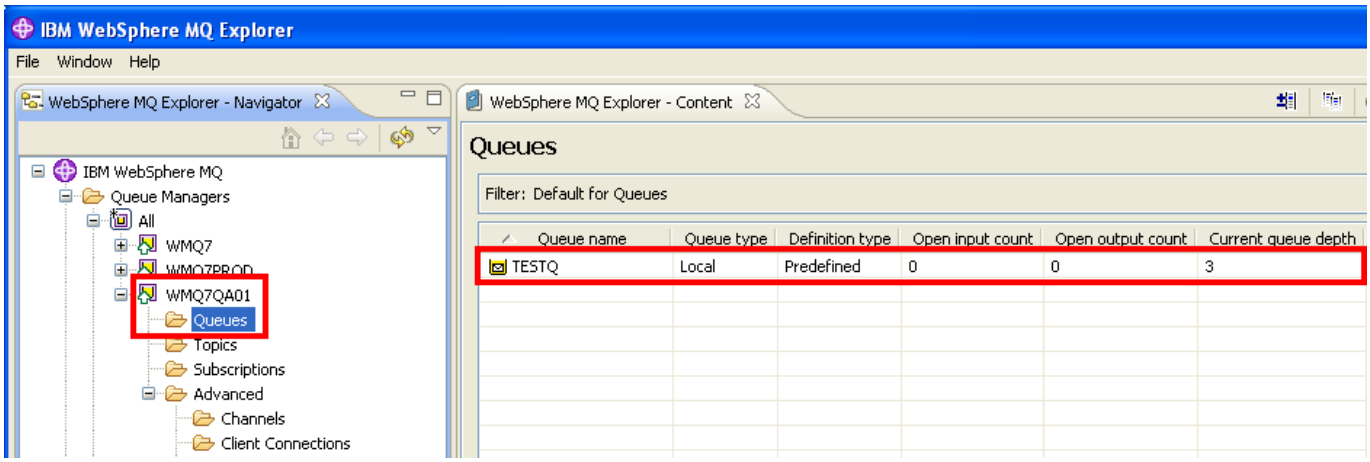
__31. Returning to the Overall channel status, you may see it go through several states (Initializing, Binding, etc) over a few seconds until it reaches **Running** status, as shown below



__32. With the channel now running, click on the **Queues** folder again for queue manager WMQ7PROD, and look at the **Current queue depth** for transmission queue **PROD.TO.QA**. Notice that now the queue is empty.



__33. In the MQ Explorer navigator pane, expand the tree below the **WMQ7QA01** queue manager, click on the **Queues** folder, and look at the **Current queue depth** for Queue **TESTQ**. Notice the queue depth of the target queue is three, indicating that your messages have been moved here from the WMQ7PROD queue manager.



__34. To confirm that these are actually the messages that you put using amqsput, you will use another sample program supplied with WebSphere MQ, called amqsget. Open another command window by clicking the icon found on the taskbar

__35. The amqsget sample program will open the specified queue and wait for messages. The format of this command is:

amqsget <q_name> <qmgr_name>

For your test, you will be getting the messages from the TESTQ local queue definition on queue manager WMQ7QA01, so enter the following command:

amqsget TESTQ WMQ7QA01

You should see the following when you enter this in the command window:

If you wish, you can go back to the amqsput window and enter some more lines of text. If you then return to the amqsget window, you will see that the messages are delivered immediately (or nearly so), because the channel is now running.

> Note: The amqsget program is designed to end after 15 seconds if no messages arrive on the queue; consequently, it's possible that the program will end before you can send your messages. If this happened, just restart the program.

When your testing is complete, terminate amqsput by just hitting the Enter key with no input text. Let amqsget terminate by waiting for the 15 second timeout to expire. Then close the two command windows.

This is the end of this portion of Lab 1

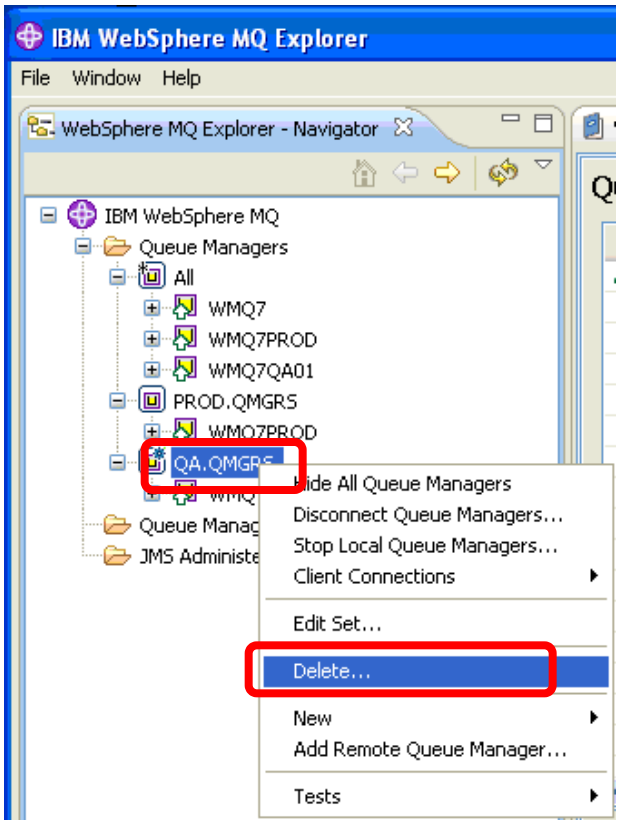
1.5 Lab Cleanup

Since you will make no further use in future labs of the last two queue managers you created, you are going to delete them.

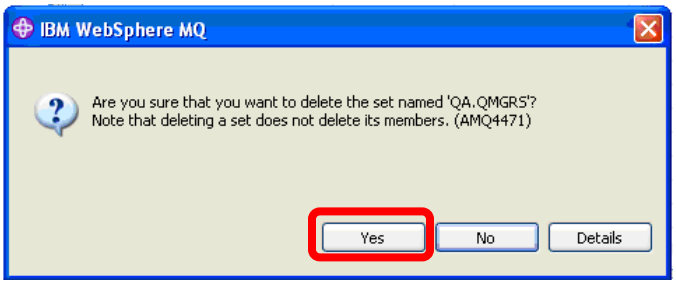
First you will remove the two Sets you created as they contain only these queue managers and would be empty after the queue managers are deleted.

Then you will stop the queue managers as you cannot delete one that is running. And then you will delete them.

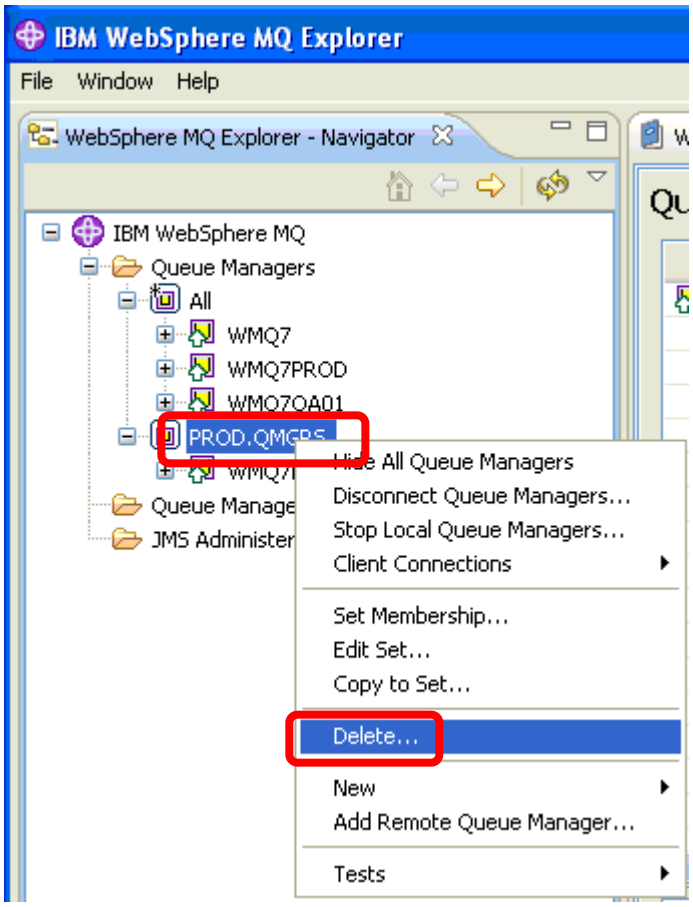
__1. Bring the MQ Explorer into view, **right-click** on the **QA.QMGRS** Set and select **Delete**.



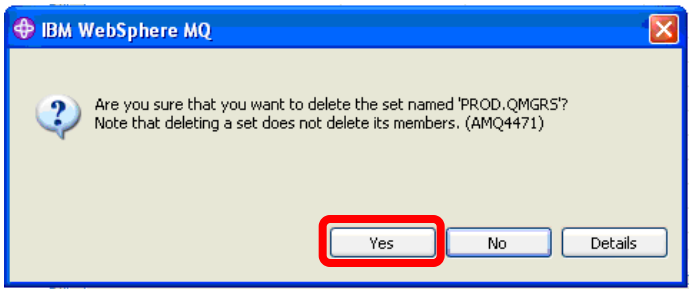
__2. Confirm the delete by clicking on the **Yes** button…



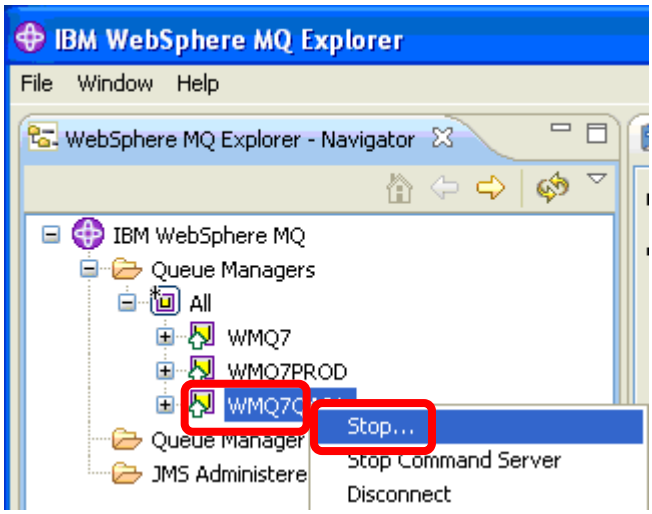
__3. **Right-click** on the **PRODS.QMGRS** Set and select **Delete**.



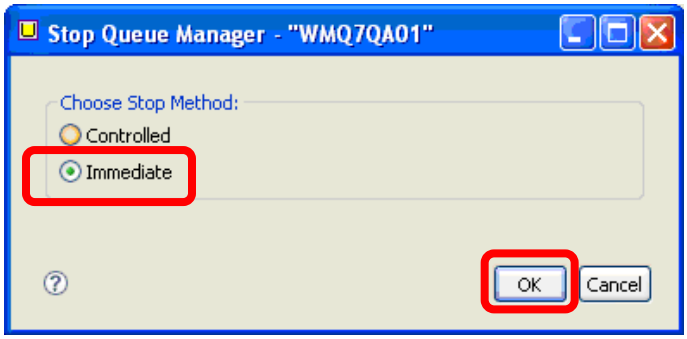
__4. Confirm the delete by clicking on the **Yes** button…



__5. **Right-click** on the **WMQ7QA01** queue manager and select **Stop…**

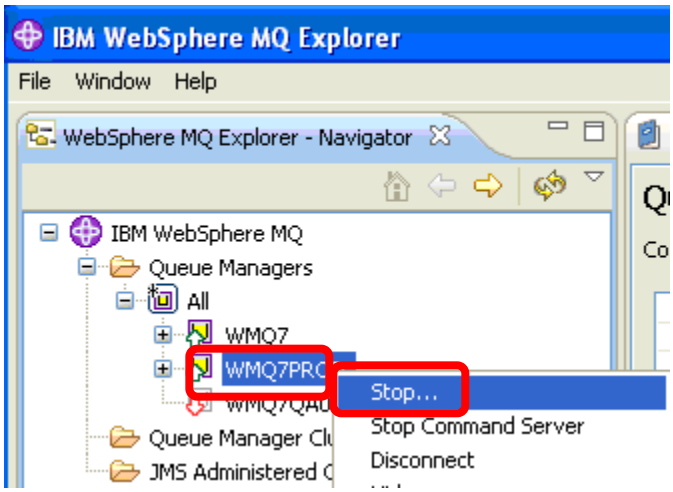


__6. Select the **Immediate** radio button and click **OK**.

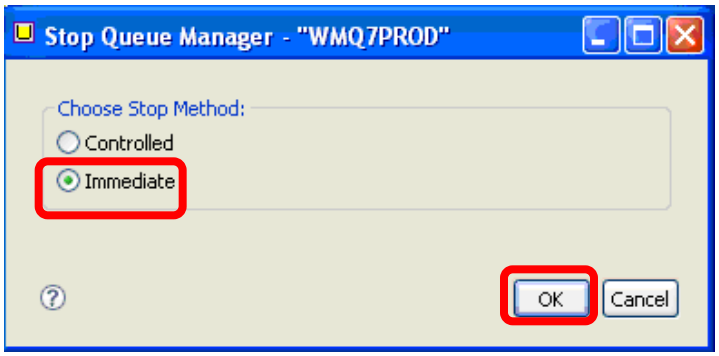


__7. The following progress panel will appear…

__8. **Right-click** on the **WMQ7PROD** queue manager and select **Stop…**

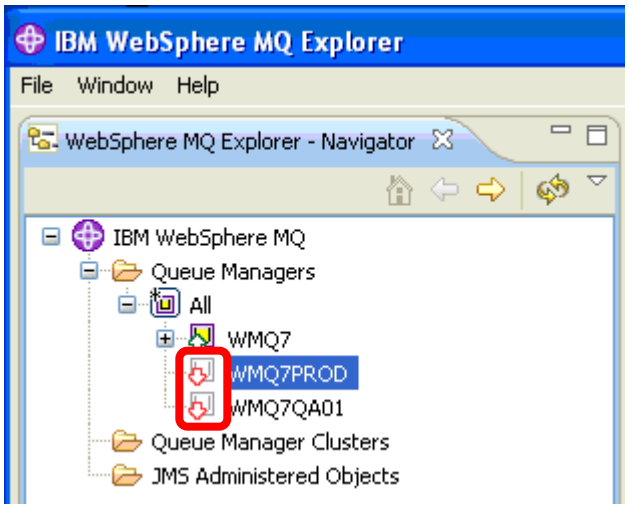


__9. Select the **Immediate** radio button and click **OK**.

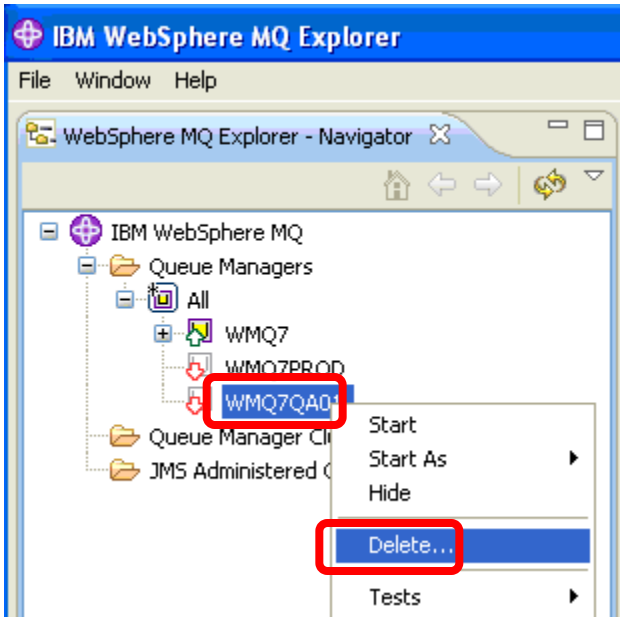


__10. The following progress panel will appear…

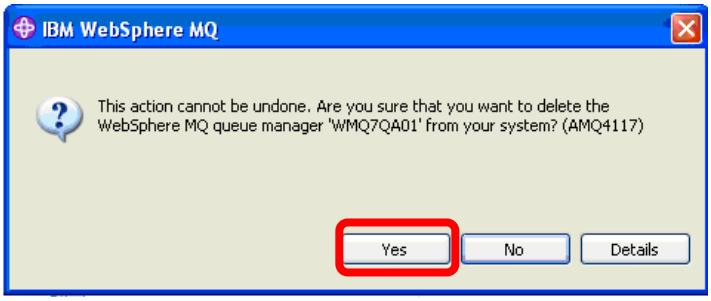
__11. Now both queue managers should be stopped.



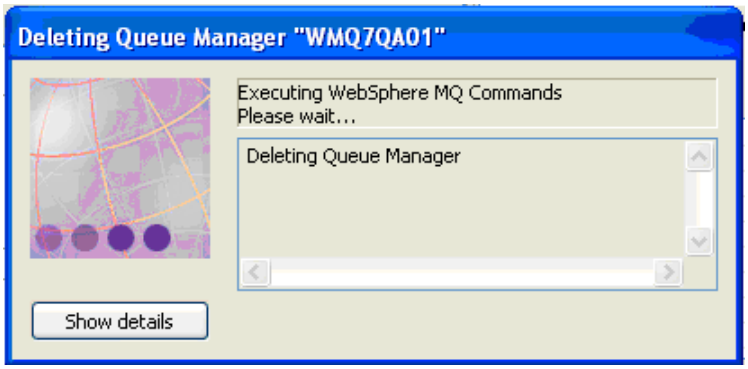
__12. Right-click on the **WMQ7QA01** queue manager and select **Delete….**



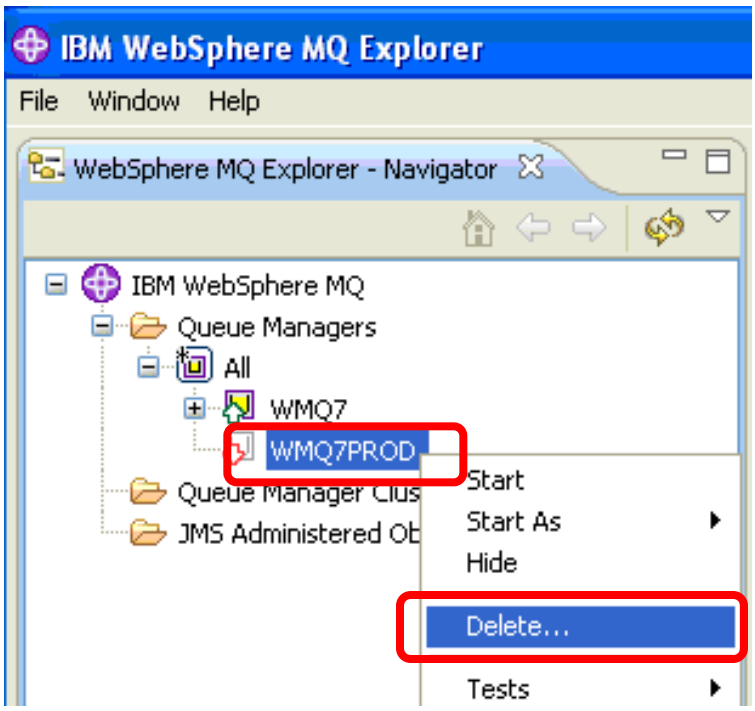
__13. Confirm the delete by clicking the **Yes** button….



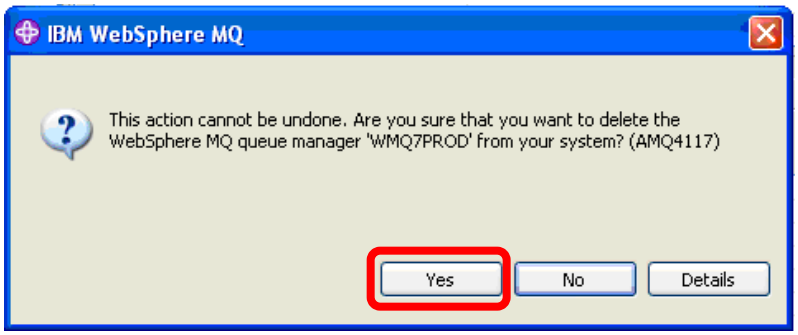
__14. The following progress panel will be displayed….



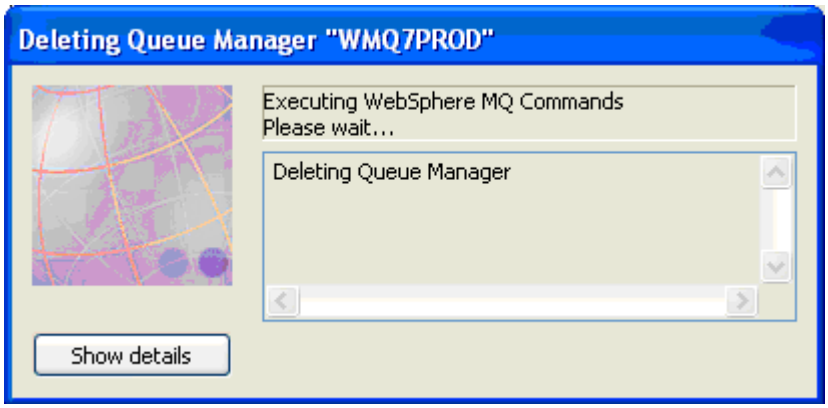
__15. Right-click on the **WMQ7PROD** queue manager and select **Delete….**



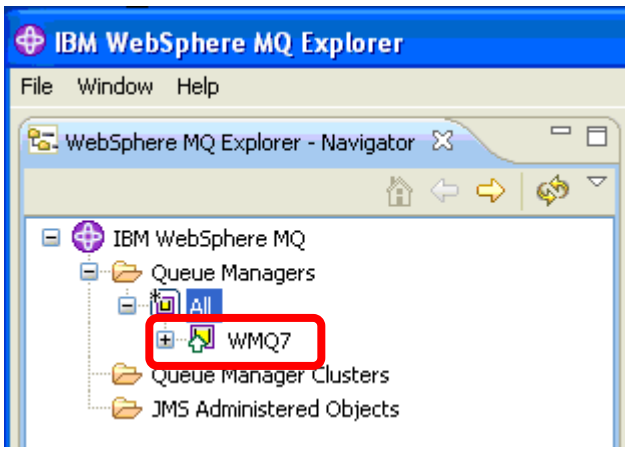
__16. Confirm the delete by clicking the **Yes** button….



__17. The following progress panel will be displayed….



__18. You should now have just a single queue manager, WMQ7



This concludes Lab 1.

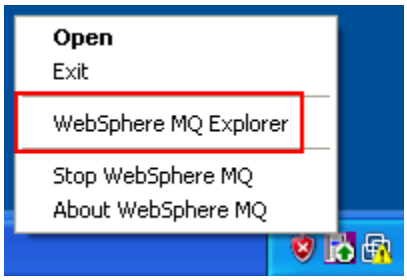
Lab 2 Configuring the WebSphere MQ JMS Provider

The purpose of this lab is to demonstrate some of the typical steps you will go through when configuring WebSphere MQ as a JMS provider. Some of the tasks that you will perform include:

- Configure the Administered objects that a JMS program typically requires. Administered objects are used to externalize **Connection Factories** and **Destinations** from the program. This allows JMS Applications to be portable between Messaging Providers by shielding the applications from provider-specific details.
- Define a **Java Naming and Directory Interface** (**JNDI) directory.** In this lab you will perform the following:
  - Create a JNDI directory
  - Populate it with Connection Factory and Destination definitions
  - Use the MQ Explorer wizard to create corresponding MQ definitions
  - Run one of the Java™ Message Service (JMS) sample programs from the command line to use those definitions to connect to MQ as a JMS provider and produce JMS messages

__1. If the MQ Explorer is not already running, you can launch it from the icon in the lower right hand corner of the screen that represents WebSphere MQ on this system.

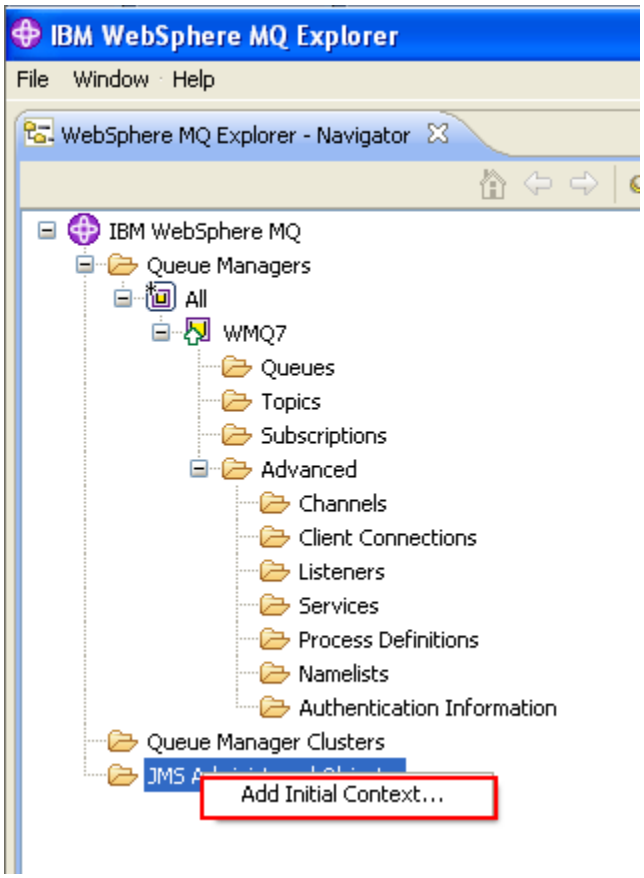
__2. Start the MQ Explorer by right-clicking on the icon and selecting **WebSphere MQ Explorer**



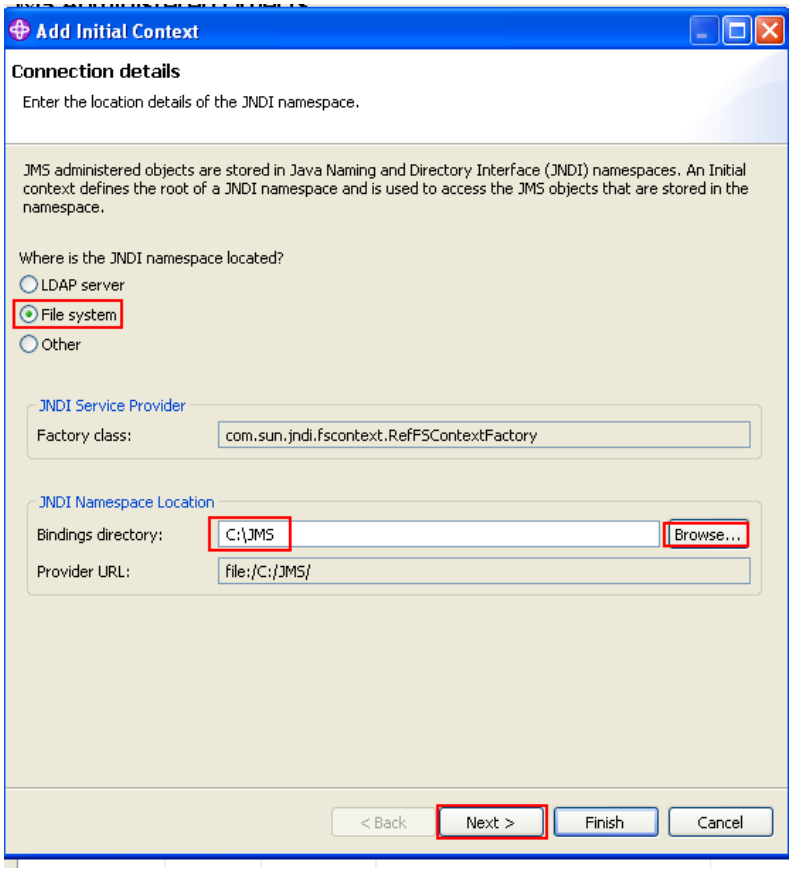
2.1 Create Administered Objects using MQ Explorer

__1. A directory has been created on your image called C:\JMS. This directory will hold the JNDI Namespace. When you create a connection to the namespace a file named .bindings will be created in this directory.

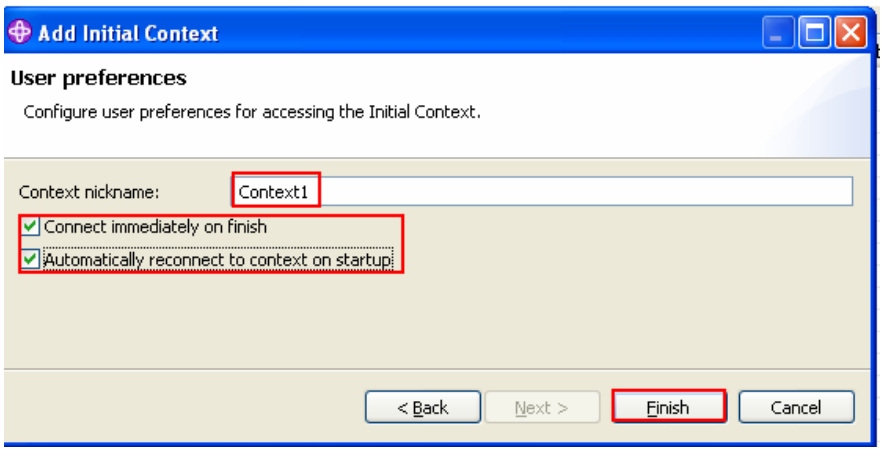
__2. Right-click on **JMS Administered Objects** in the Navigator pane, and select **Add Initial Context…**



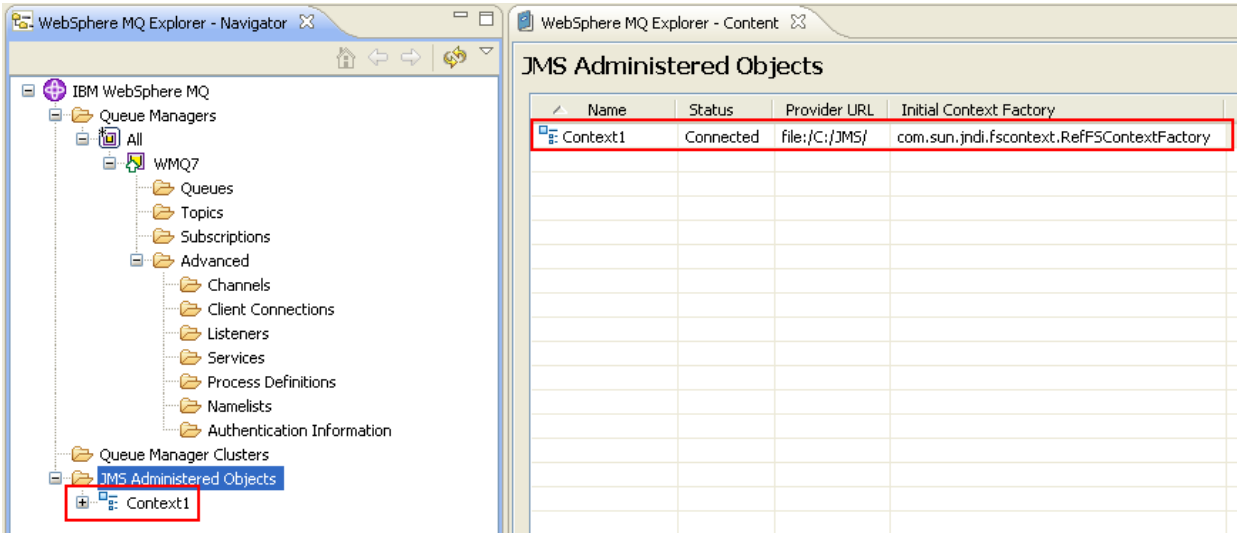
__3. On the **Connection details** screen, click on **File System**. Then click on **Browse** to navigate to the directory called **C:\JMS**. Click Next.



__4. On the **'User preferences'** screen, enter the Context nickname **Context1**. Note this can be a name of your choice but for this lab you will use **Context1**. This name will not be used elsewhere. Check both the **Connect immediately on finish** and **Automatically reconnect to context on startup** checkboxes, and then click **Finish**.



__5. The newly created initial context is displayed in the list.

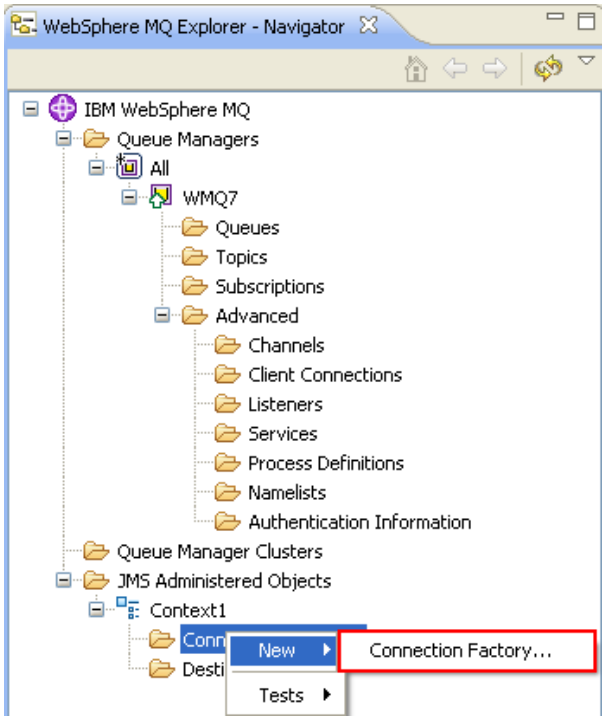


This concludes this portion of Lab 2.

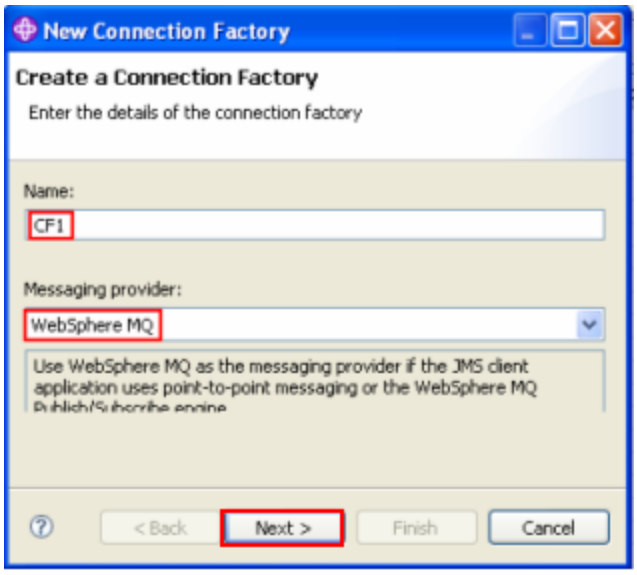
2.2 Create a connection factory for WebSphere MQ

A connection factory is the mechanism used by JMS to manage connections between your JMS application and the JMS Provider. You will now define a connection factory in the JNDI namespace.

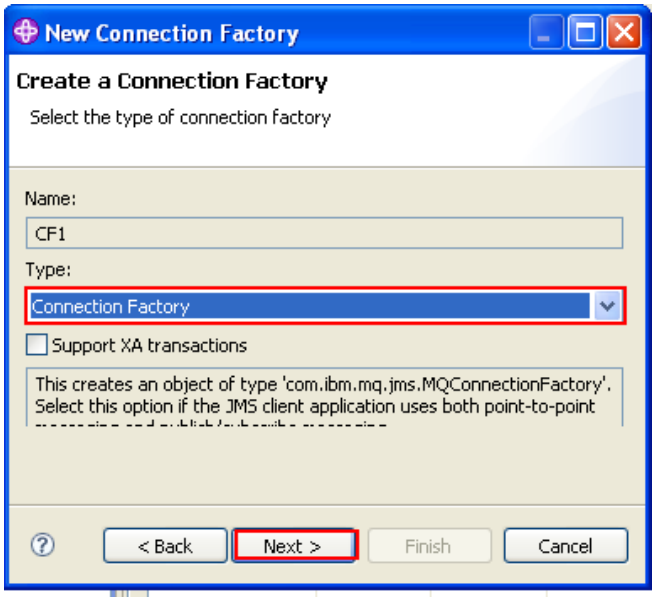
__1. Expand the context name you just created. Right-click **Connection Factories.** Select **New-> Connection Factory.**



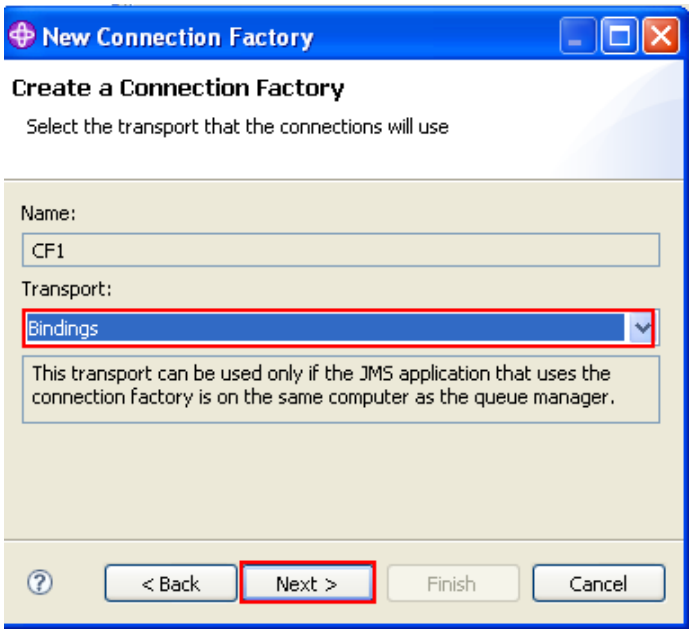
__2. Enter the name **CF1**. Note this can be any name of your choice but for this lab you will use CF1. This will be required when running the program so you might want to make note of it. Accept **WebSphere MQ** as the messaging provider and then click **Next >**



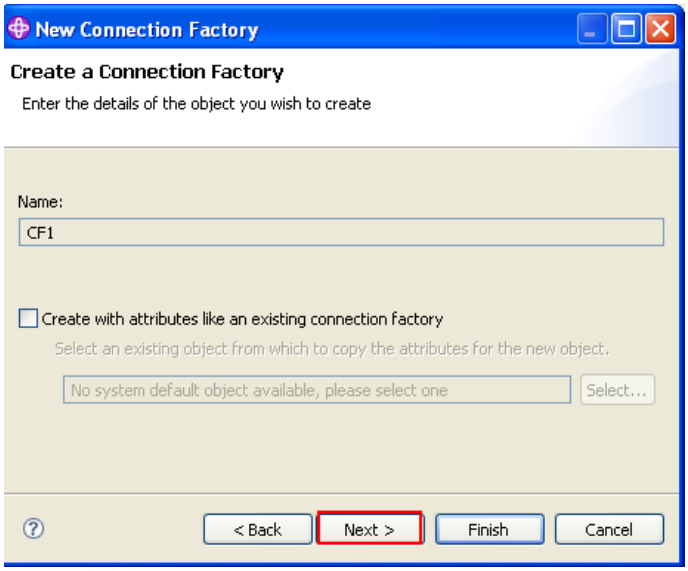
__3. On the next screen, accept the Type as **Connection Factory** and leave **Support XA transactions** unchecked. Click on **Next >**.



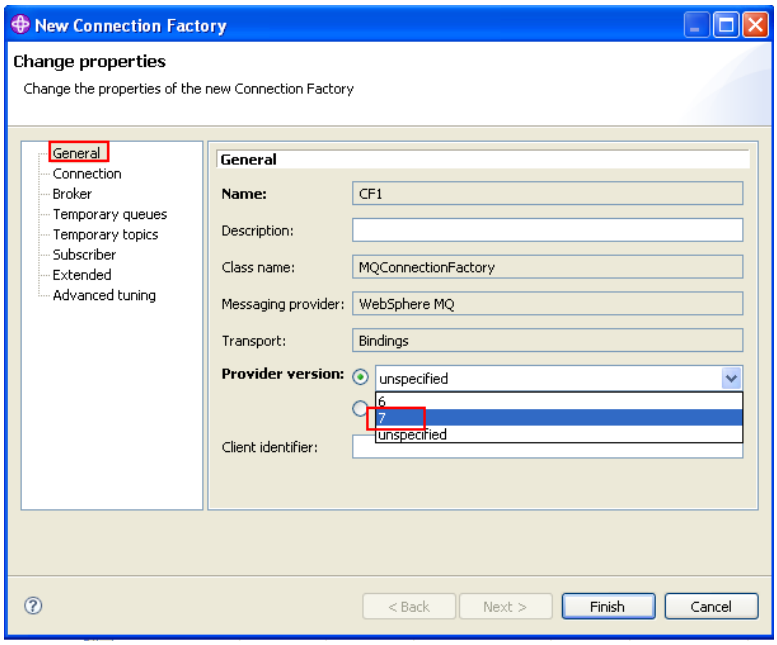
__4. On the next screen, accept **Bindings** as the Transport given we will run the JMS Application on the same machine as the Queue Manager. Click **Next >**.



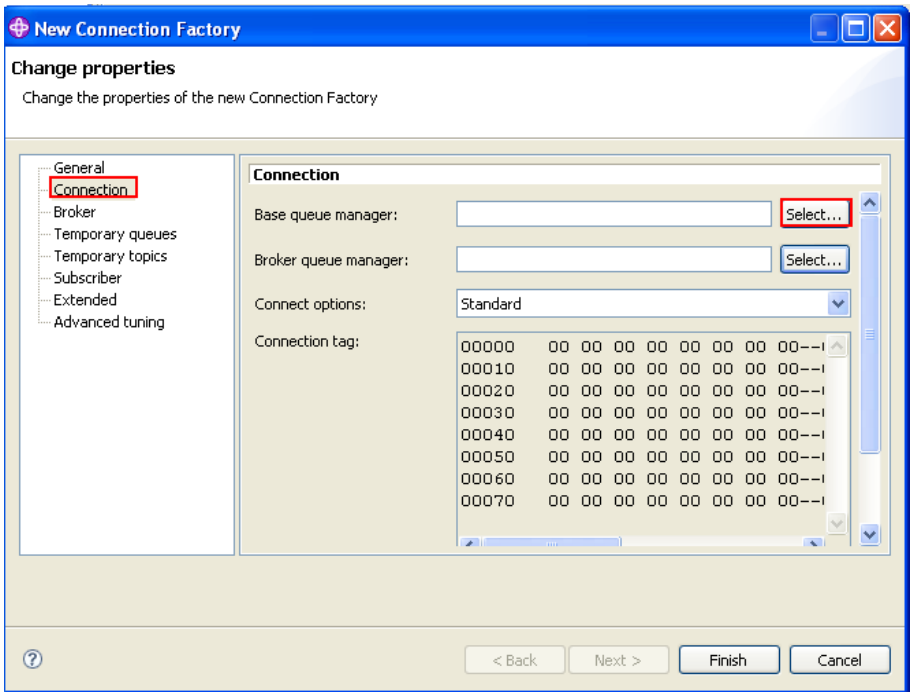
__5. On the next screen, leave 'Create with attributes like an existing connection factory' unchecked, and click **Next >**.



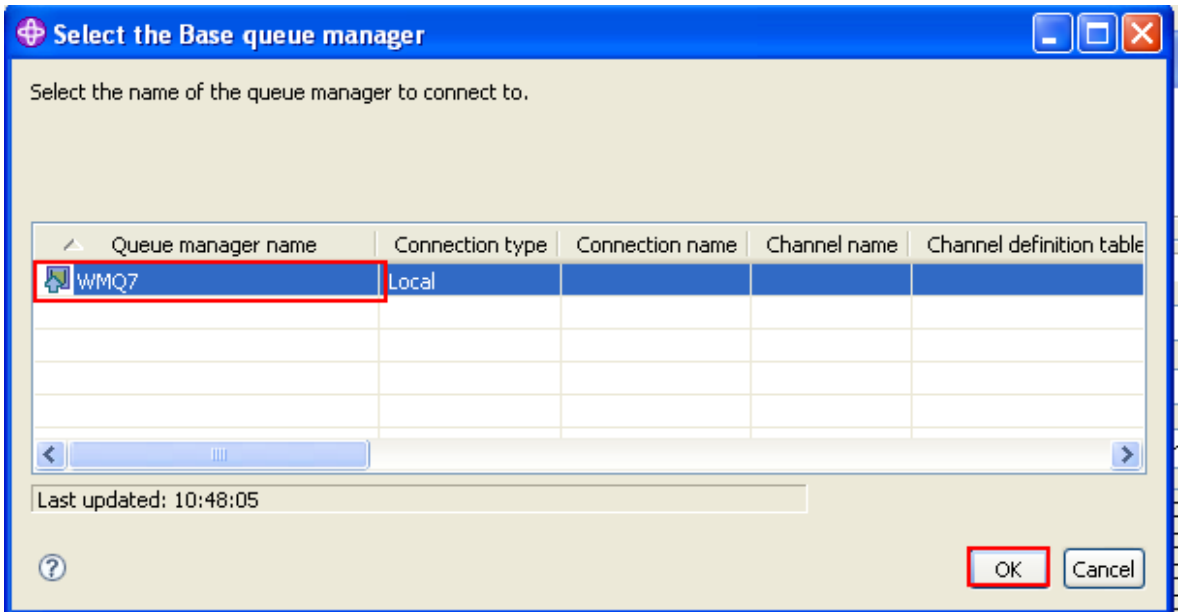
__6. This will open the **New Connection Factory** property sheet. Select **General** on the left hand menu. Use the Provider version **'7**'. This represents WebSphere MQ V7. Selecting this will enable JMS programs using this connection factory to utilize the new features of WebSphere MQ V7.



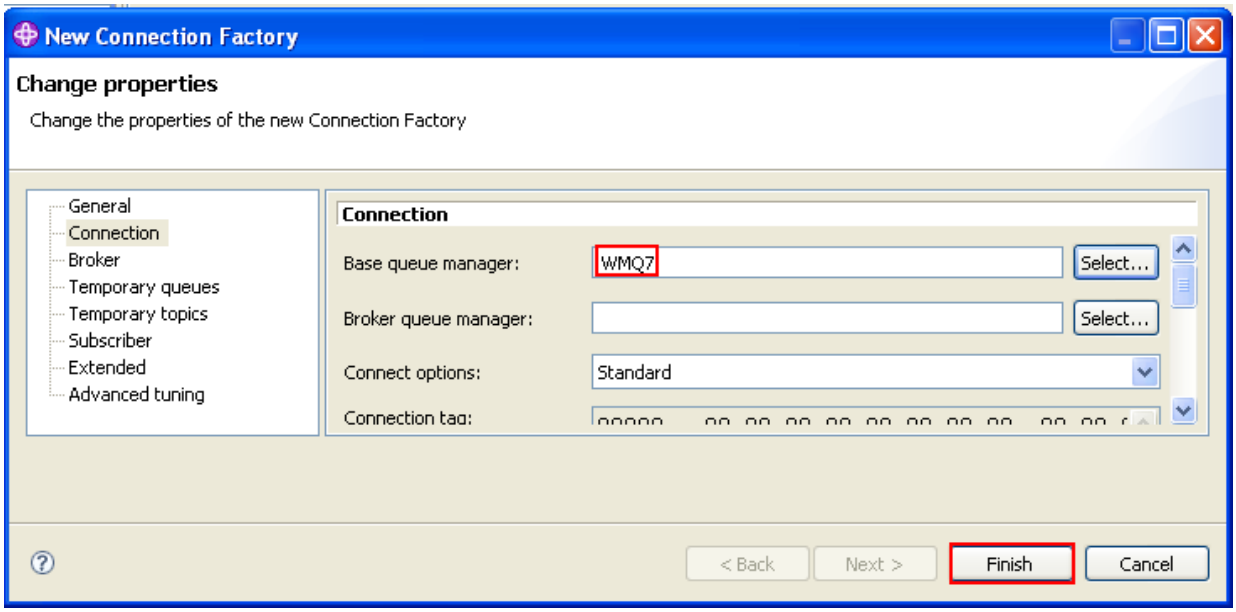
__7. Select **Connection** on the left hand menu. Click on the "**Select**" button for **'Base queue manager'**



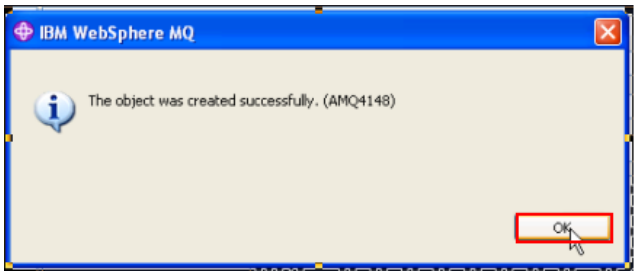
__8. Select the previously-defined Queue Manager **WMQ7** and then click **"OK".**



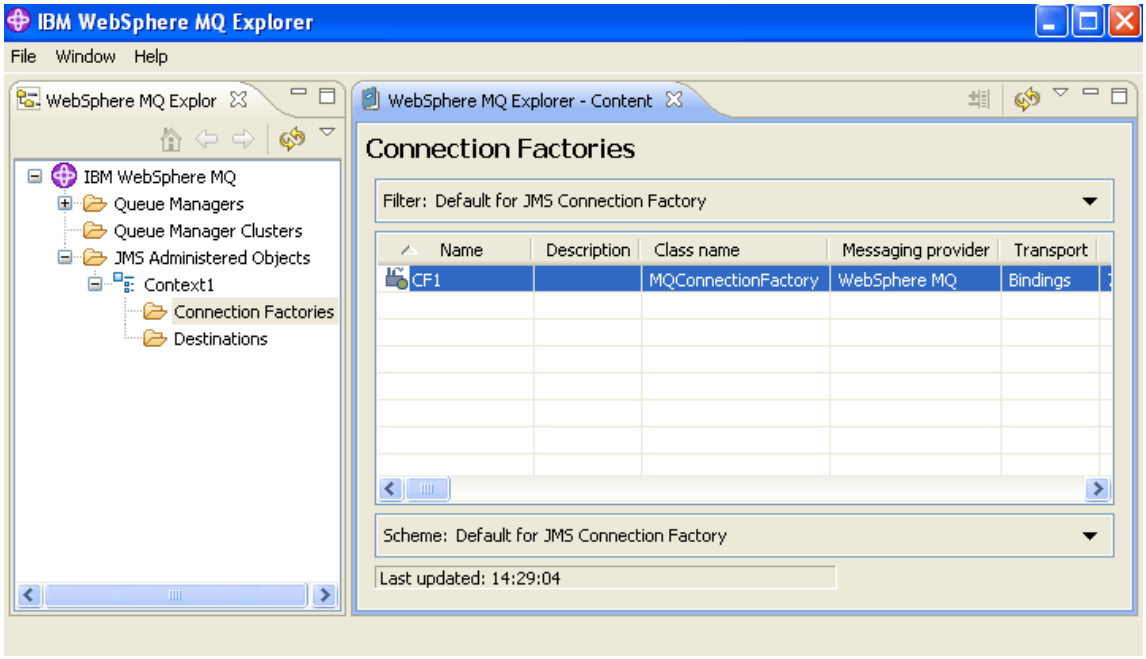
__9. Click **Finish** to create the connection factory.



__10. Click **OK** to dismiss the confirmation box



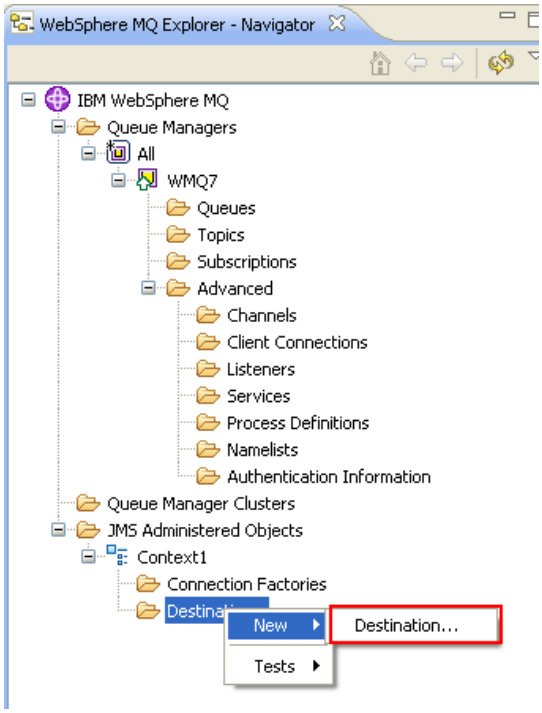
__11. Observe that Connection Factory **CF1** now appears in the Content list.



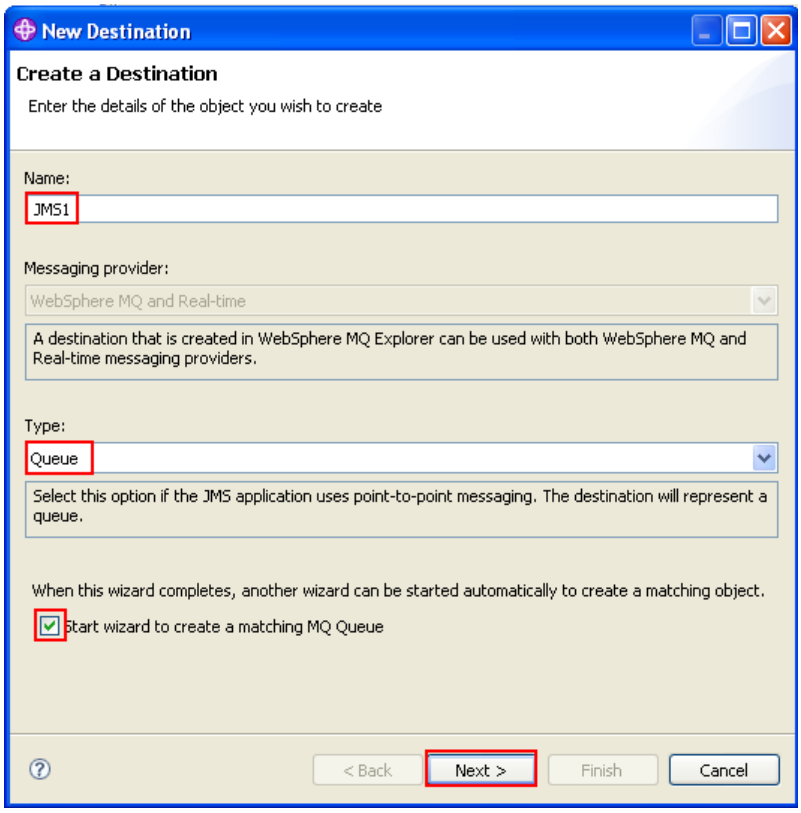
This concludes this portion of Lab 2.

2.3 Create a Destination for the JMS Application to put a message to.

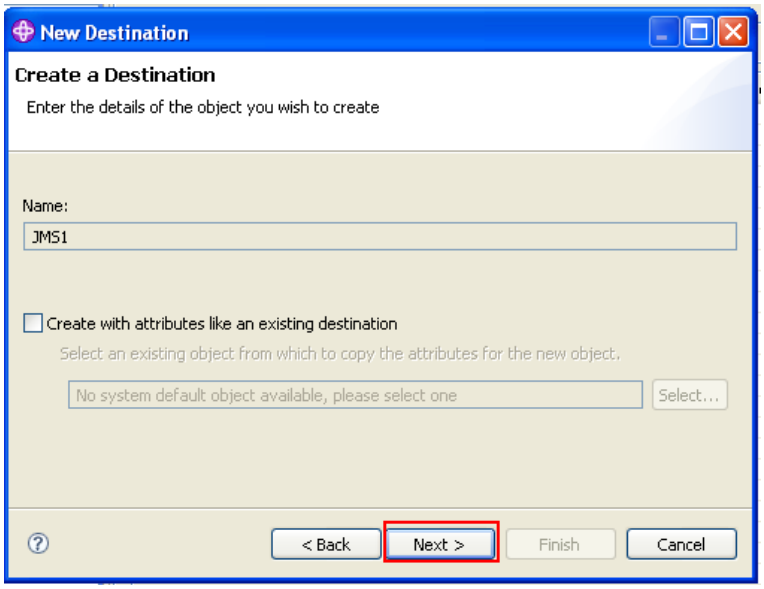
__1. Right-click ‘Destinations’ and click New → Destination…



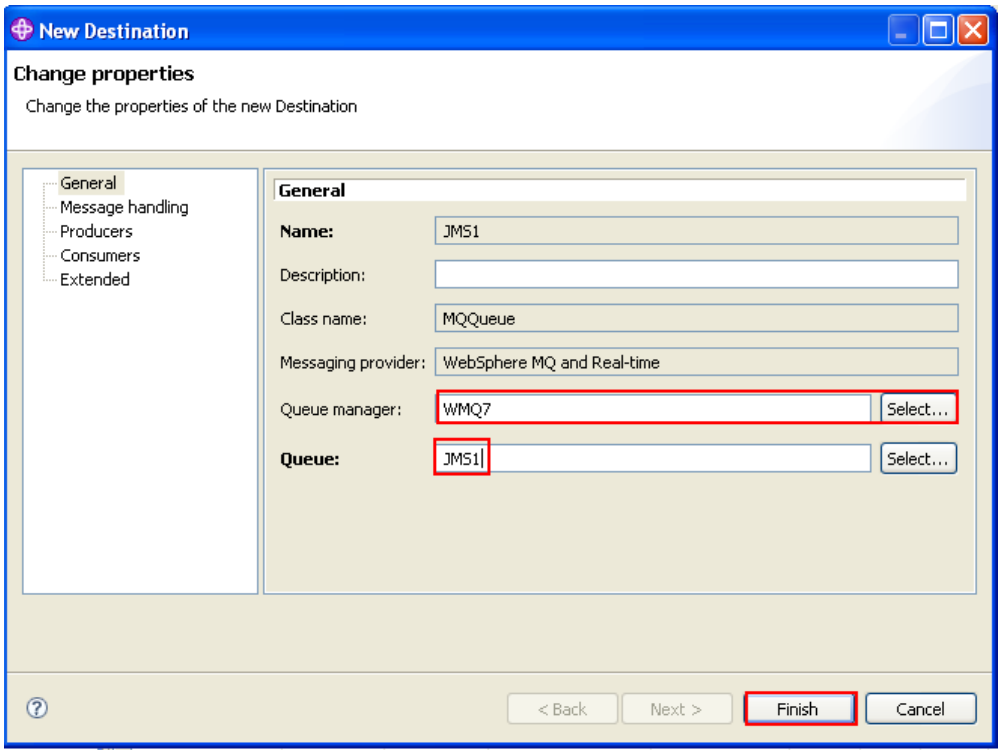
__2. Enter the name **JMS1**. Note this can be any name of your choice but for this lab you will use **JMS1**. Leave the Type as '**Queue**' and ensure '**Start wizard to create a matching MQ Queue'** is checked. This will create a corresponding WebSphere MQ Queue. The Queue that you create will be used to verify that messages are successfully written using the sample program. Click **Next >.**



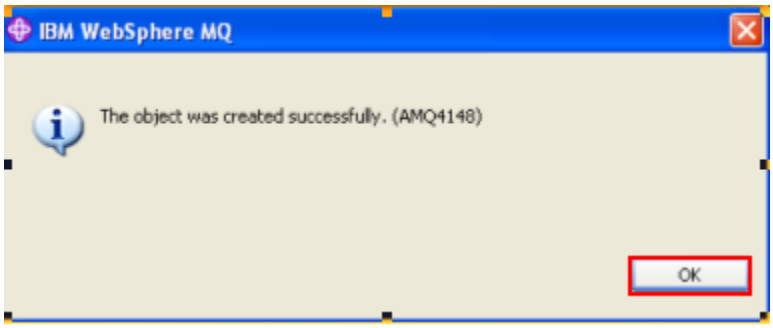
__3. Accept the defaults on this next screen and click **Next >**.



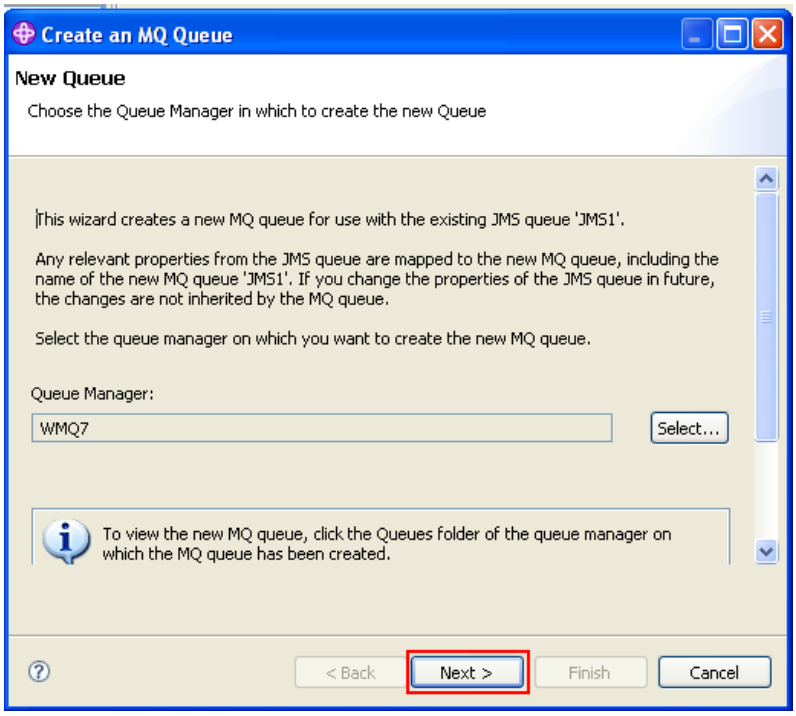
__4. On the change properties screen you can set some MQ-specific properties of the destination. Select **WMQ7** as the Queue Manager and define a Queue with the same name as the JMS Destination. Note the Queue name can be any name of your choice but for this lab you will use **JMS1**. This will be created as part of the upcoming wizard. Click **Finish** to launch the MQ configuration wizard.



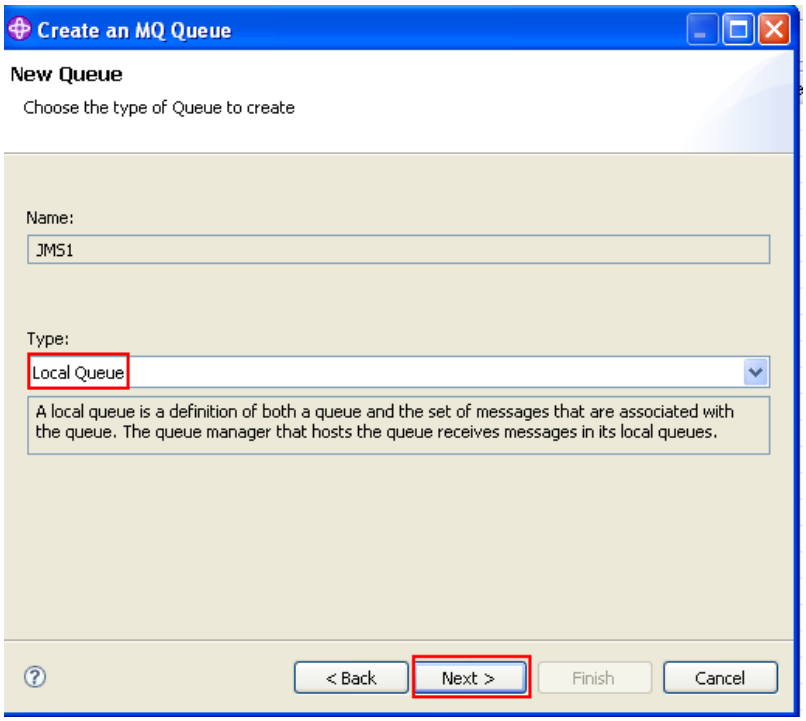
__5. Click **OK** to dismiss the confirmation screen.



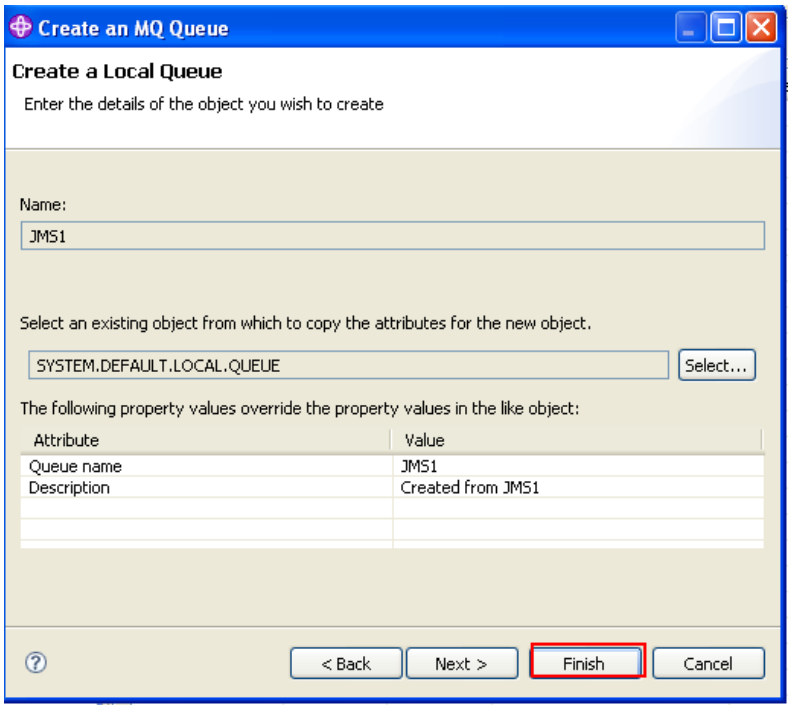
__6. The **Create an MQ Queue** wizard will start automatically. Click **Next >**.



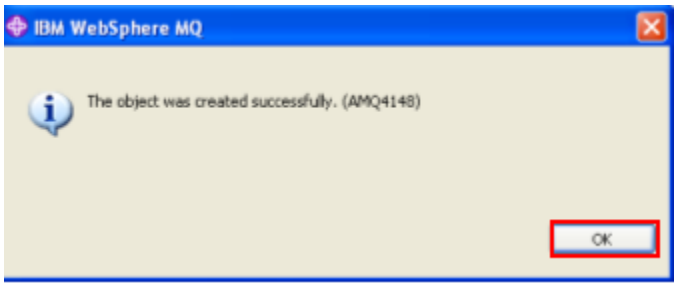
__7. Accept the default '**Local Queue**' for Type and then click **Next >**



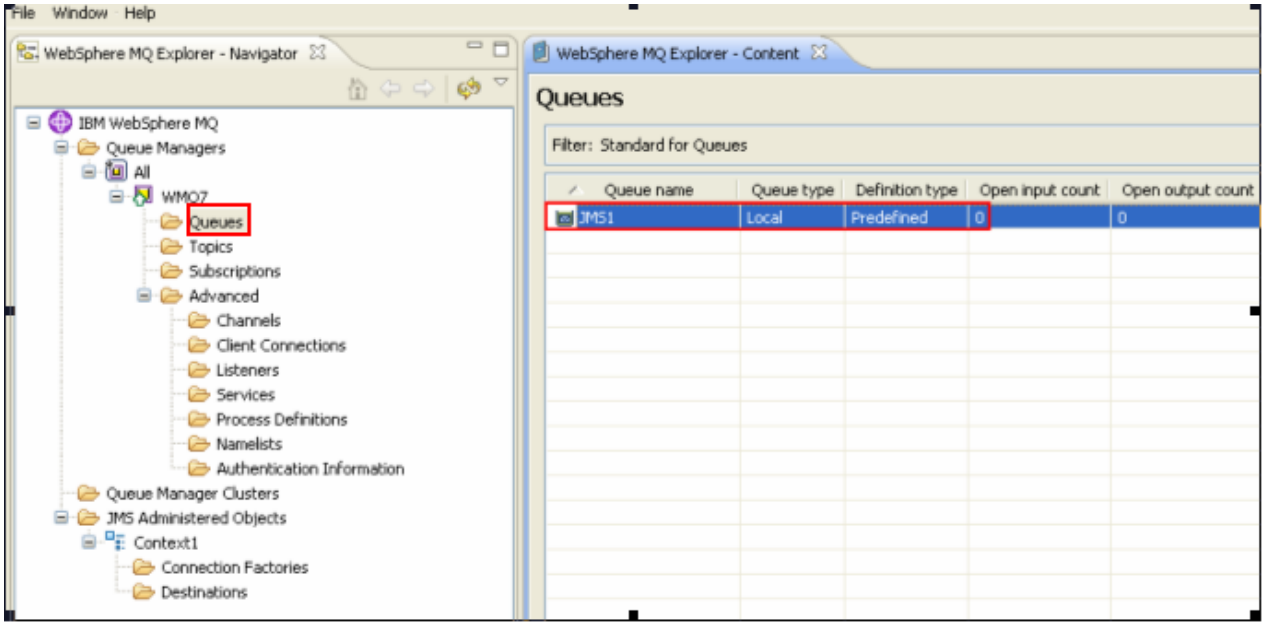
__8. Click **Finish** to create the MQ queue.



__9. Click **OK** to close the confirmation prompt.



__10. Click on the Queues folder under WMQ7 in the MQ Explorer Navigator pane, and look at the Content pane to confirm that an MQ local queue called **JMS1** has been created.



This concludes this portion of Lab 2.

2.4 Writing a JMS message using a Java sample program

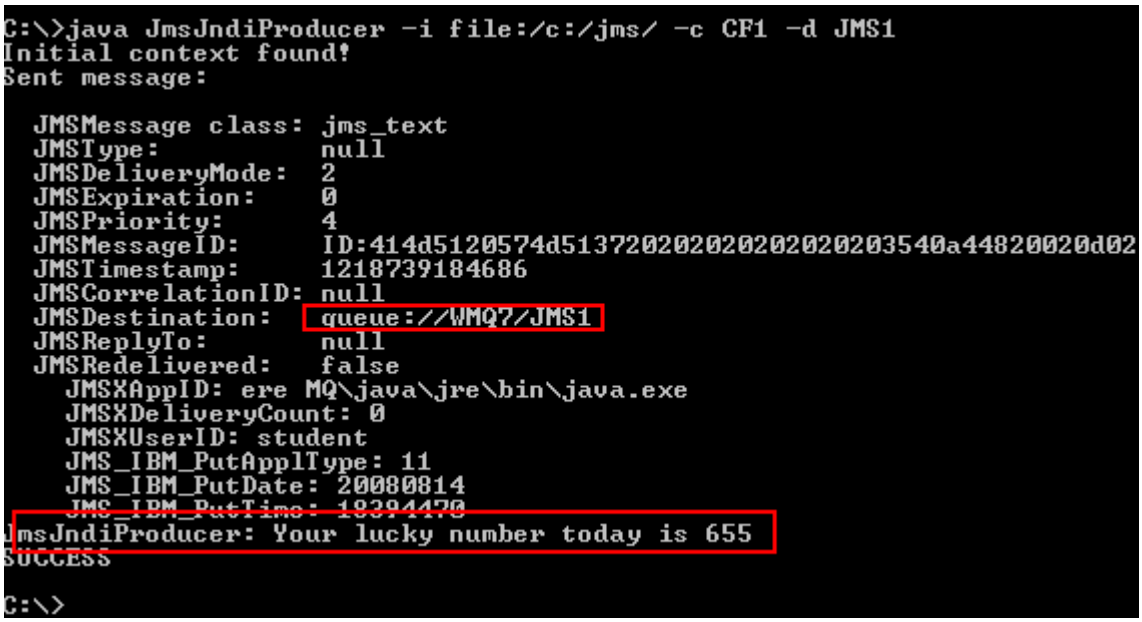
Some very good code samples ship with WebSphere MQ V7 and you will use one of them to create a JMS message on an MQ JMS queue.

__1. Open a command prompt. A shortcut for this is in the start menu.

__2. Type in the following command:

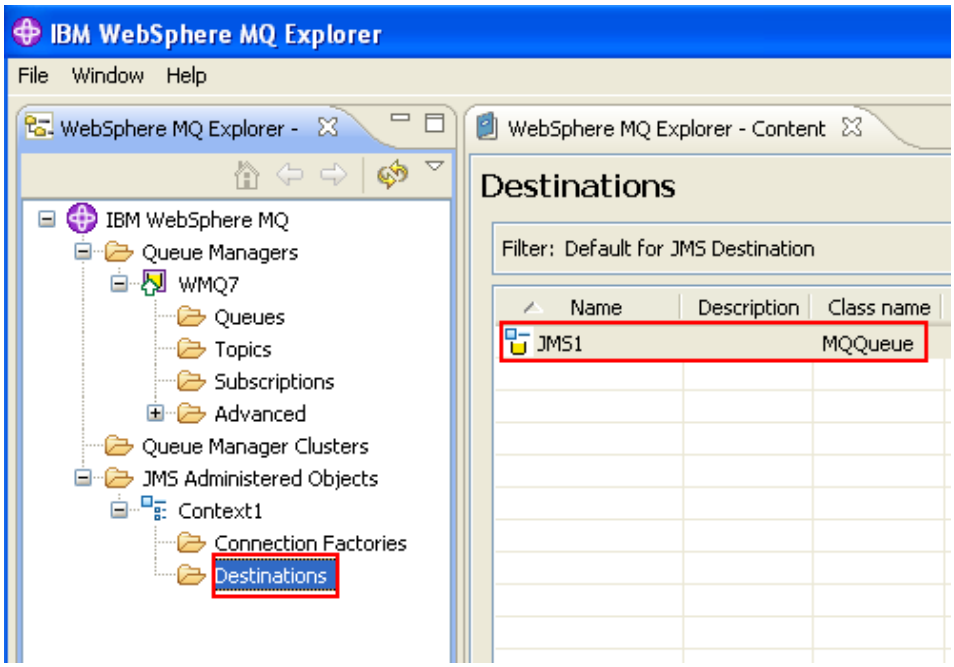
java JmsJndiProducer -i file:/c:/jms/ -c CF1 -d JMS1

This will run the Java JMS sample program JmsJndiProducer. The **"-i"** argument points the program to the location of your JNDI directory. The **"-c CF1"** identifies the connection factory for your test queue manager, and the **"-d JMS1"** identifies the JMS destination queue.

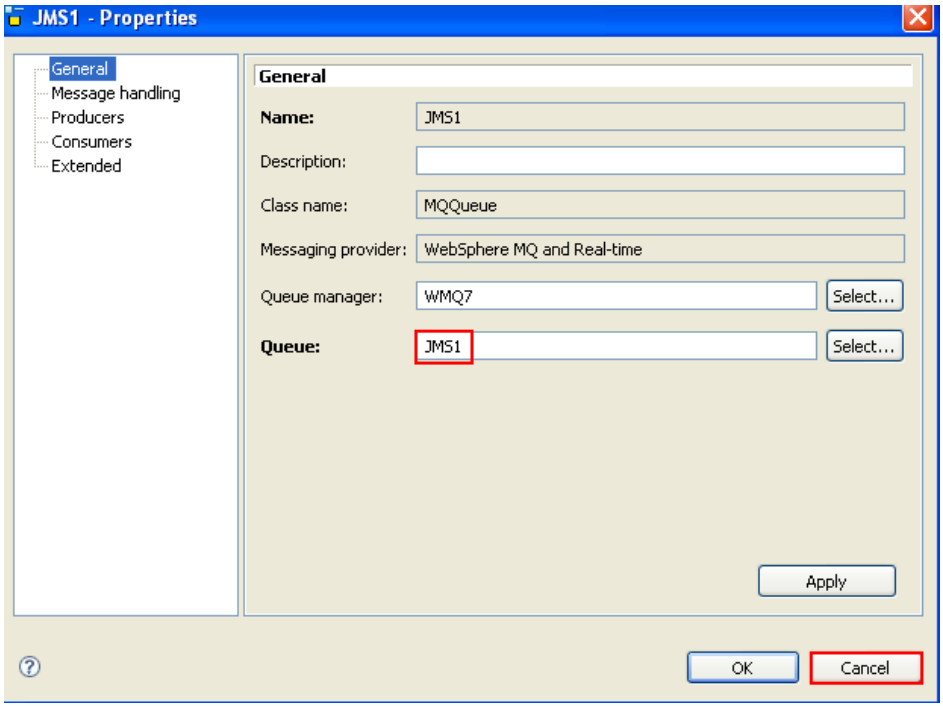


Having run this program, you should now have a message in the **JMS1** queue. To see whether you do, you can look for the message that was produced by this program using the WebSphere MQ Explorer. Switch back to the MQ Explorer window

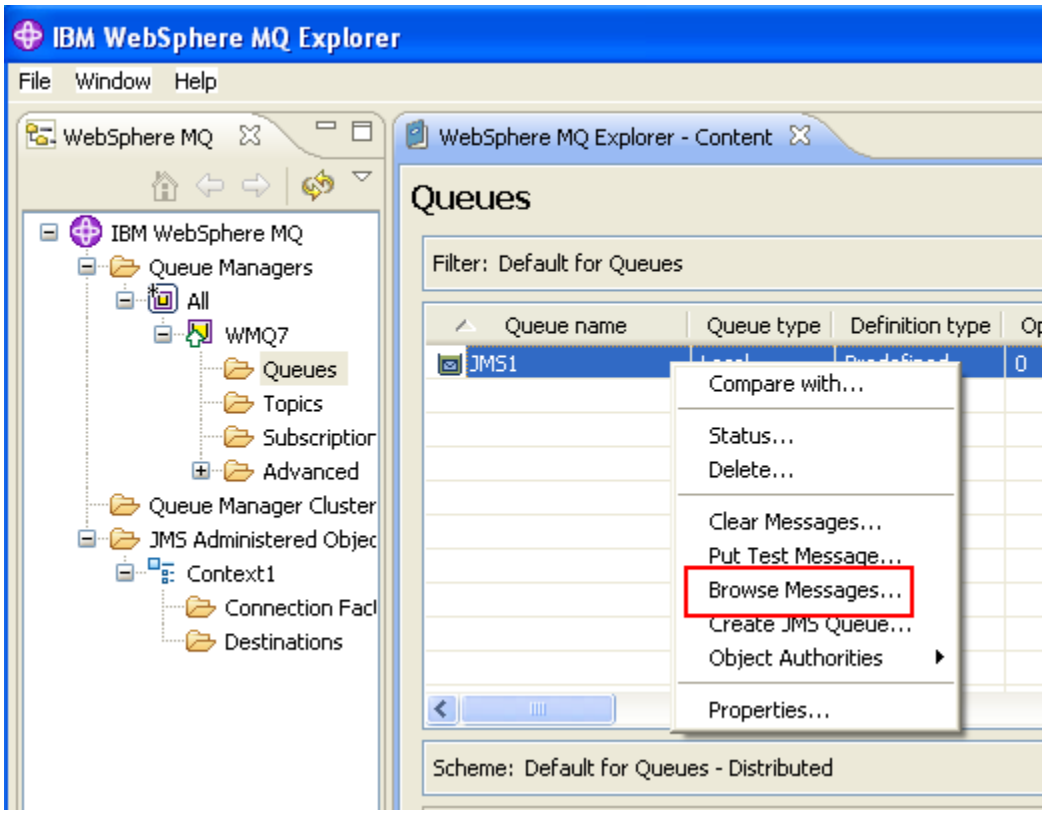
__3. In the MQ Explorer Navigator pane, locate the folder called **JMS Administered Objects**. Click on the **Destinations** folder beneath it, and in the Content pane you should see your JMS Destination **JMS1**. Double-click on the JMS1 destination to find the MQ Queue that is associated with it.



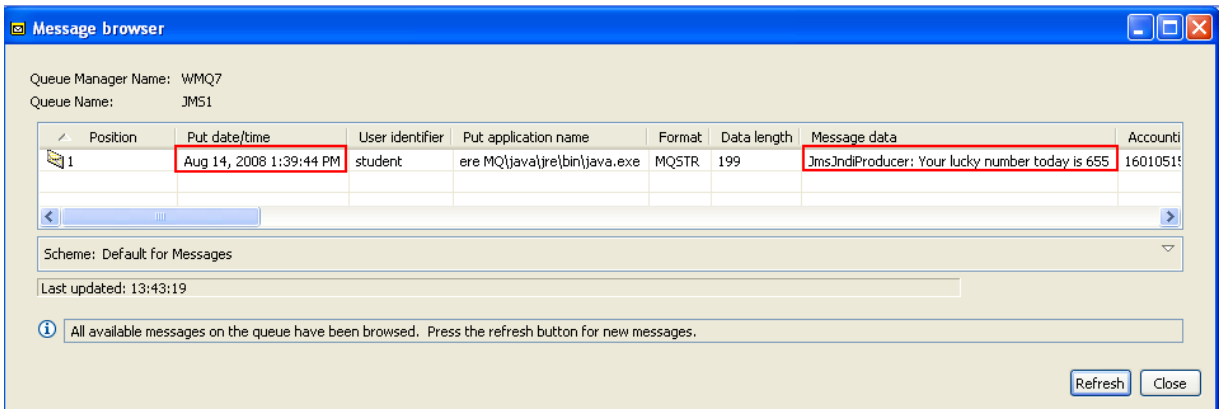
__4. Recall that MQ queue **JMS1** is associated with the JMS destination **JMS1**. Click on **Cancel.**



__5. Right-click on the JMS1 MQ queue object and then select **Browse Messages…**



__6. The message data column in the display should match the message text that you wrote with the JmsJndiProducer program.



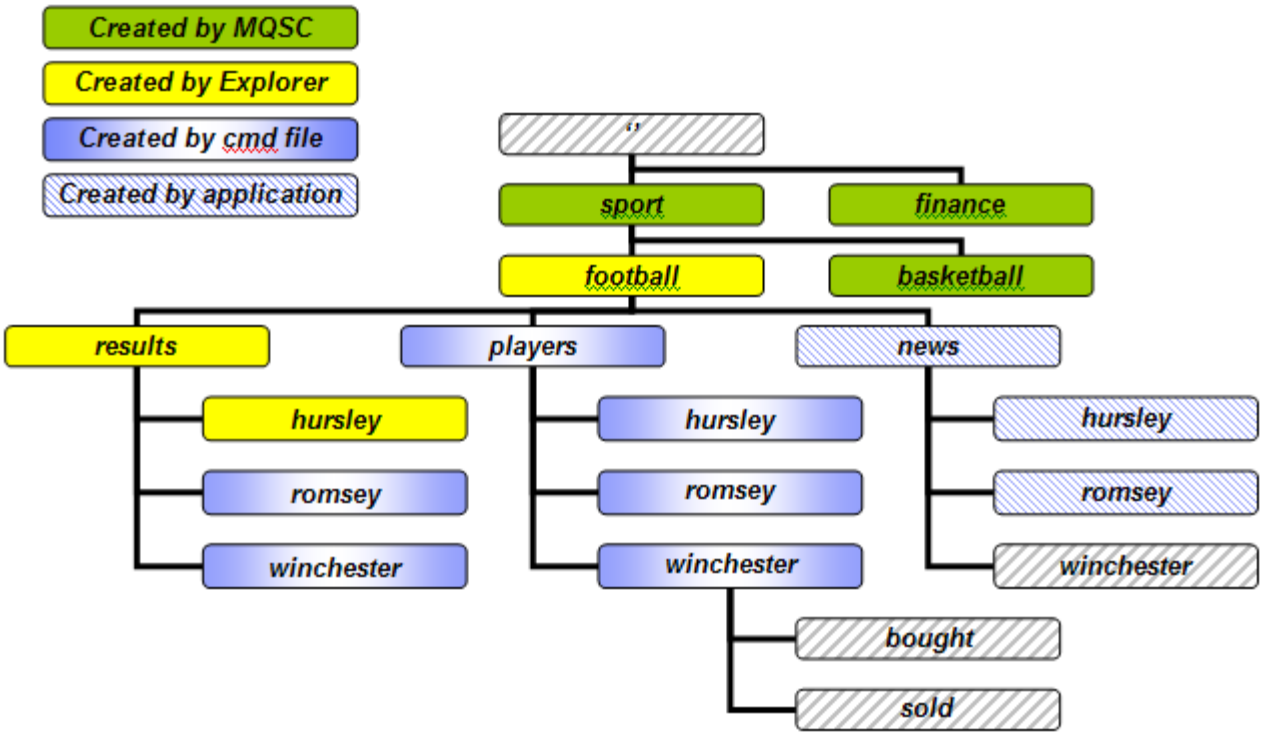
This concludes Lab 2.

Lab 3 Publish / Subscribe Administration Lab

3.1 Lab Overview

This lab will demonstrate the administration of Publish / Subscribe-related objects using the WebSphere MQ Explorer. Also, you will use the MQ Explorer Test Publication and Test Subscription features to demonstrate publishing and subscribing to TOPIC strings.

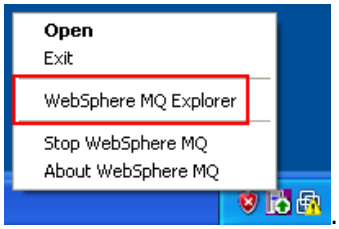
You will be working with the hierarchical structure of TOPIC STRINGS represented in the picture below.



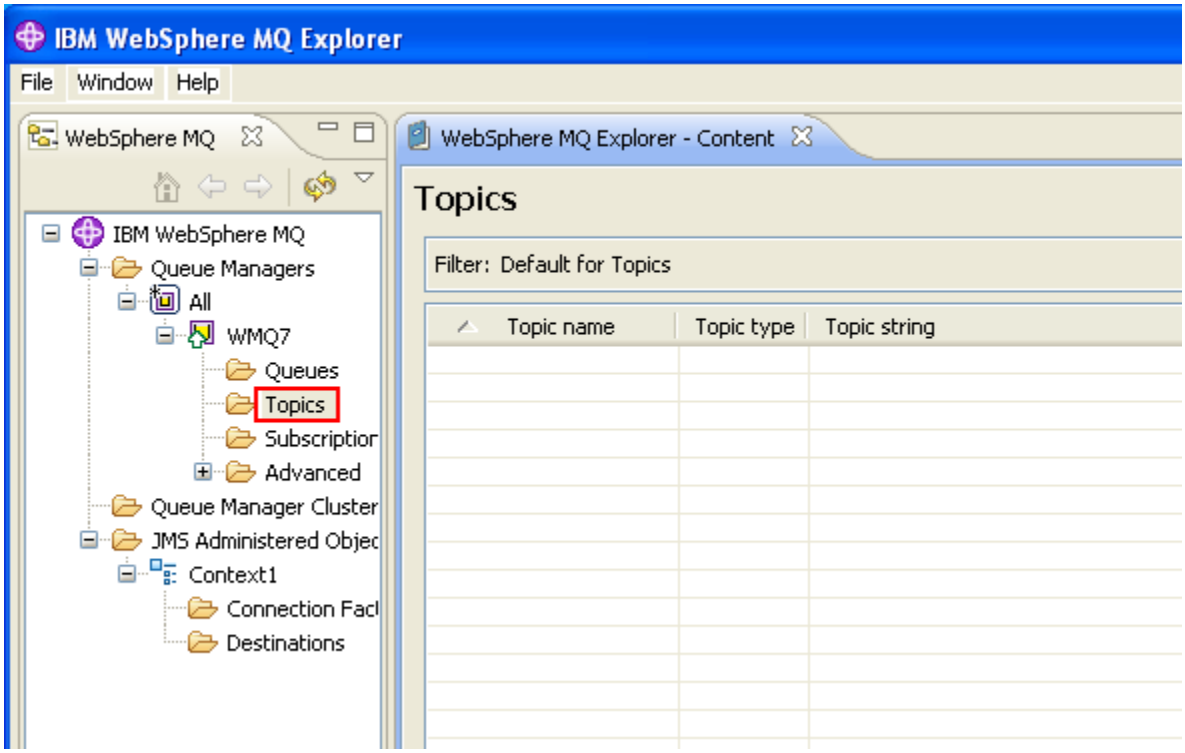
3.2 Using MQ Explorer to create and display information

__1. If the MQ Explorer is not already running, you can launch it from the icon in the lower right hand corner of the screen that represents WebSphere MQ on this system.

__2. Start the MQ Explorer by right-clicking on the icon and selecting **WebSphere MQ Explorer**



__3. Click on the **Topics** folder in the Navigator pane (left side). Notice that you have no topic objects defined yet! We have provided a script for this purpose.

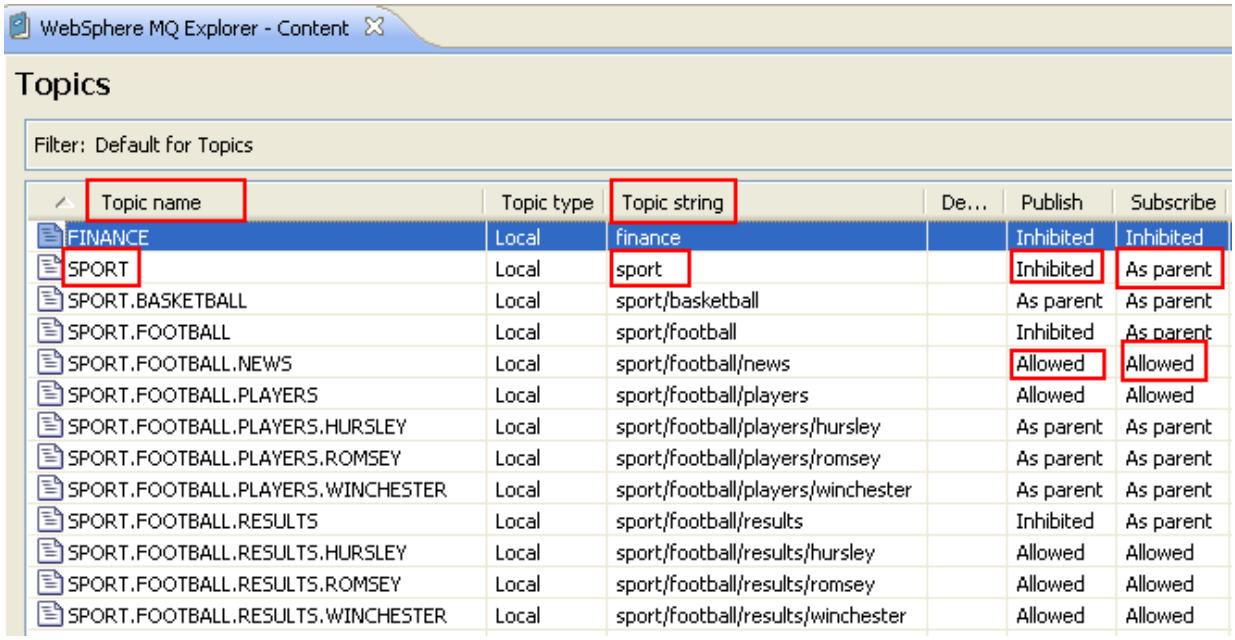


__4. To run the provided script, find the shortcut called **PubSub Lab Setup** on the Windows® desktop. Double-click the shortcut to run the script.

__5. The script should run very quickly, leaving the following command window open. Verify that the command was successful. Then press Enter to close the command window.

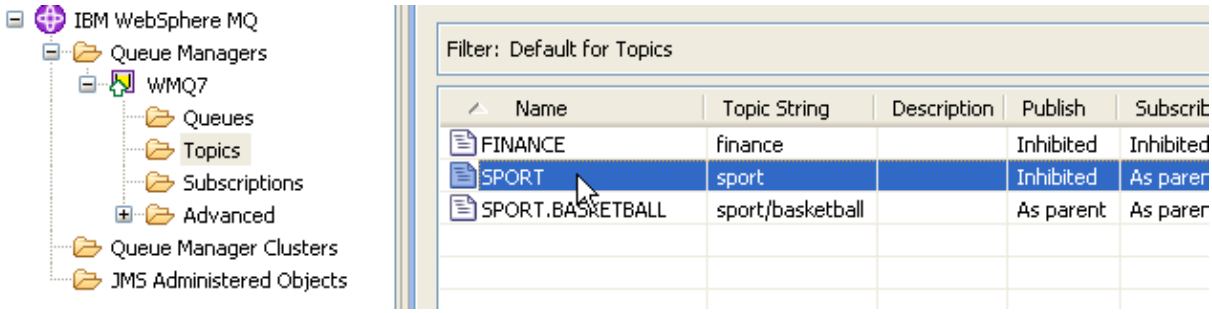
```
        :         SUB(ASPARENT)
AMQ8690: WebSphere MQ topic created.
        :
     7 : DEFINE TOPIC(SPORT.FOOTBALL.NEWS) +
       :         TOPICSTR('sport/football/news') +
       :         REPLACE +
       :         PUB(ENABLED) +
       :         SUB(ENABLED)
AMQ8690: WebSphere MQ topic created.
7 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.

C:\Student\Lab_PubSubAdmin\resources>pause
Press any key to continue . . .
```

__6. You should now see the following WebSphere MQ Topic objects displayed in the Content pane in the MQ Explorer. Observe the Topic objects (under the Topic name column) and their corresponding Topic string values. Also observe the Publish and Subscribe enablement status on the right.

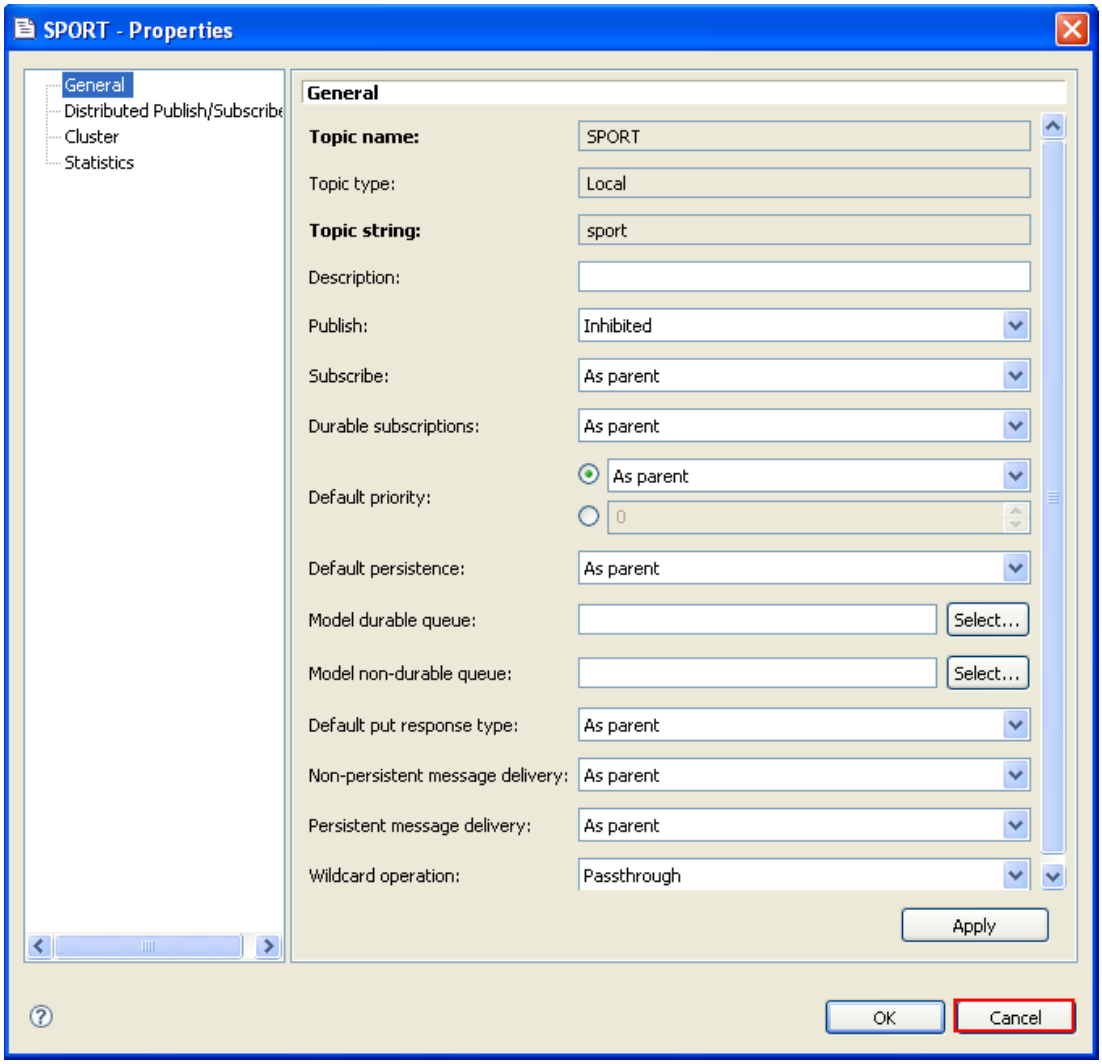


Topic name	Topic type	Topic string	De...	Publish	Subscribe
FINANCE	Local	finance		Inhibited	Inhibited
SPORT	Local	sport		Inhibited	As parent
SPORT.BASKETBALL	Local	sport/basketball		As parent	As parent
SPORT.FOOTBALL	Local	sport/football		Inhibited	As parent
SPORT.FOOTBALL.NEWS	Local	sport/football/news		Allowed	Allowed
SPORT.FOOTBALL.PLAYERS	Local	sport/football/players		Allowed	Allowed
SPORT.FOOTBALL.PLAYERS.HURSLEY	Local	sport/football/players/hursley		As parent	As parent
SPORT.FOOTBALL.PLAYERS.ROMSEY	Local	sport/football/players/romsey		As parent	As parent
SPORT.FOOTBALL.PLAYERS.WINCHESTER	Local	sport/football/players/winchester		As parent	As parent
SPORT.FOOTBALL.RESULTS	Local	sport/football/results		Inhibited	As parent
SPORT.FOOTBALL.RESULTS.HURSLEY	Local	sport/football/results/hursley		Allowed	Allowed
SPORT.FOOTBALL.RESULTS.ROMSEY	Local	sport/football/results/romsey		Allowed	Allowed
SPORT.FOOTBALL.RESULTS.WINCHESTER	Local	sport/football/results/winchester		Allowed	Allowed

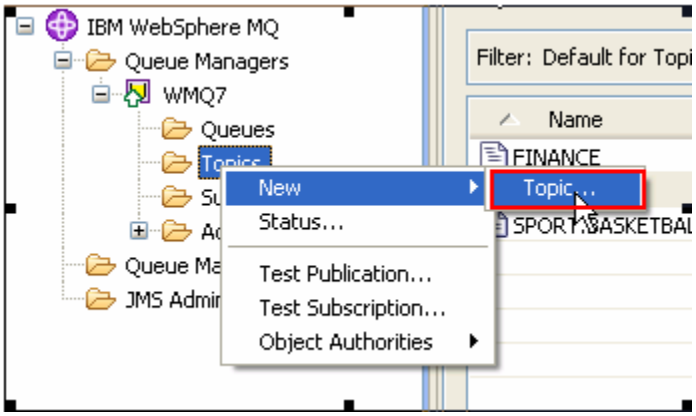
__7. Double-Click on the **SPORT** Topic object.



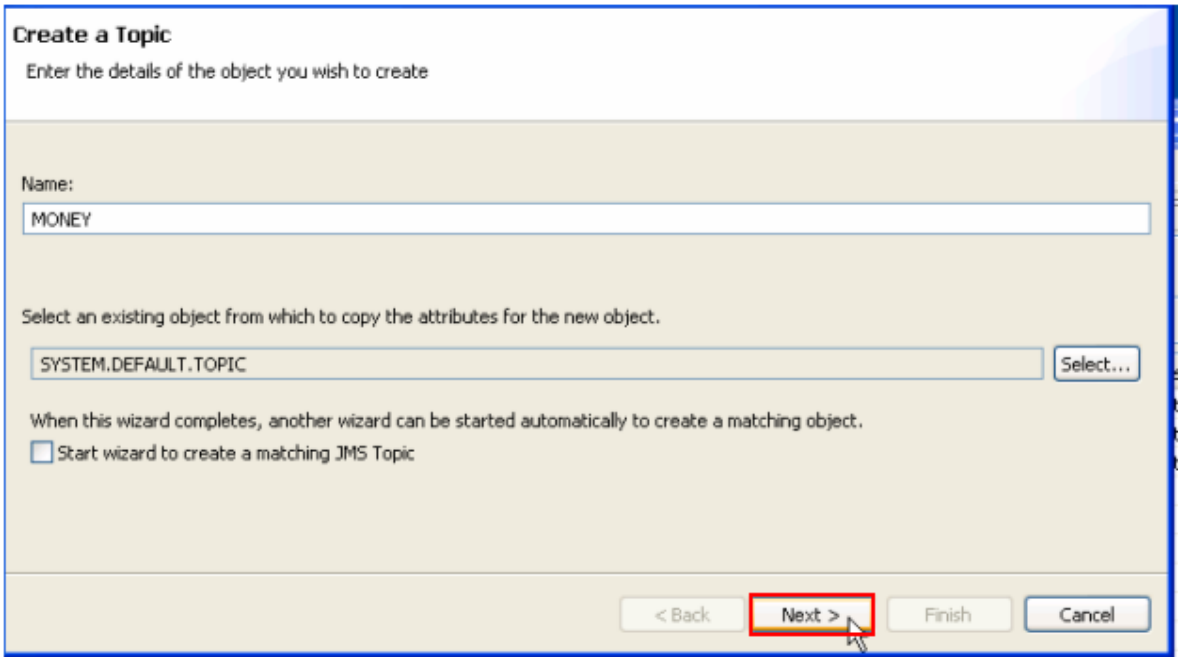
__8. Observe the various properties of the Topic object. Explore the various values available on the pull-downs if you wish. Close the window by clicking "**Cancel**".



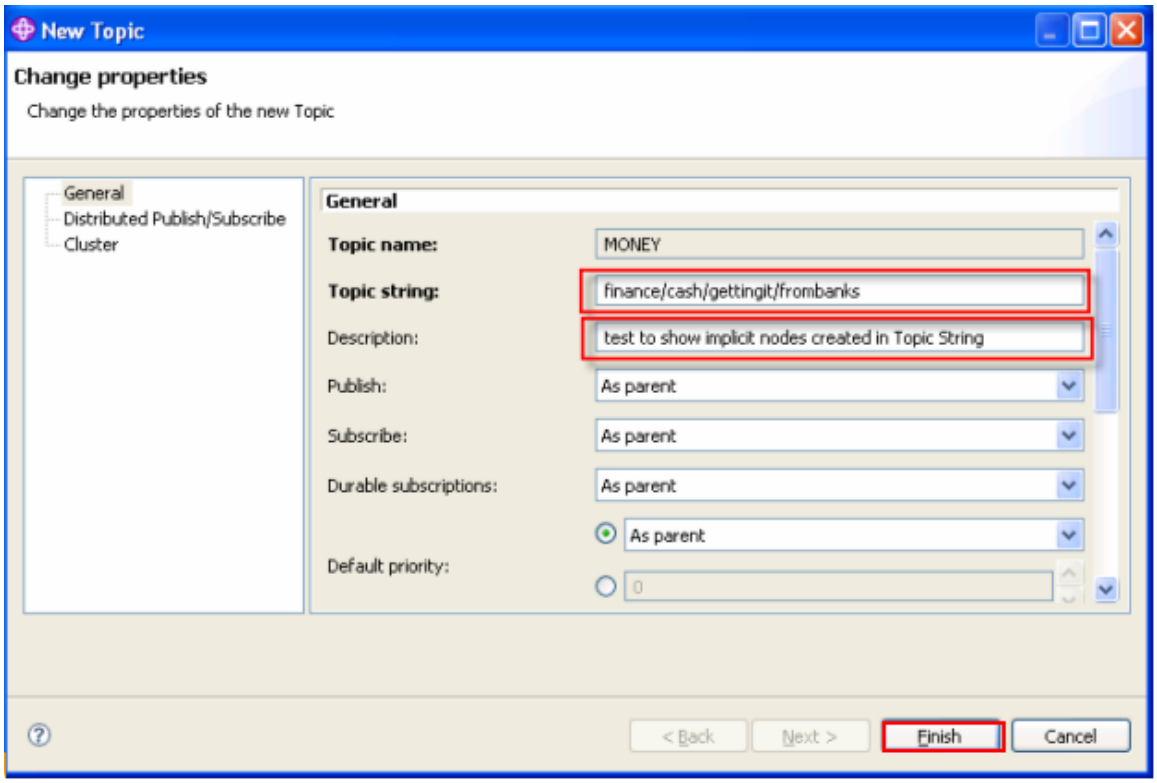
__9. From the Navigation pane select "**Topics**" under queue manager WMQ7. Right-click on **Topics** and then select **New→Topic** from the context menu.



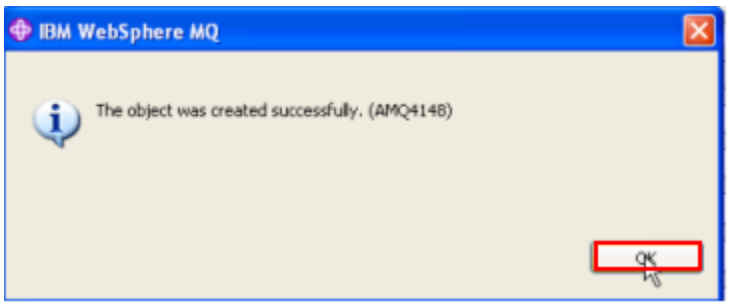
__10. Enter **MONEY** in the Name field; allow the other fields to default. Then click "**Next**".



__11. Enter **“finance/cash/gettingit/frombanks**” (without the quote marks) in the **Topic string** field. Enter a description in the description field and then click “**Finish**”



__12. Close the confirmation box by clicking “**OK**”.

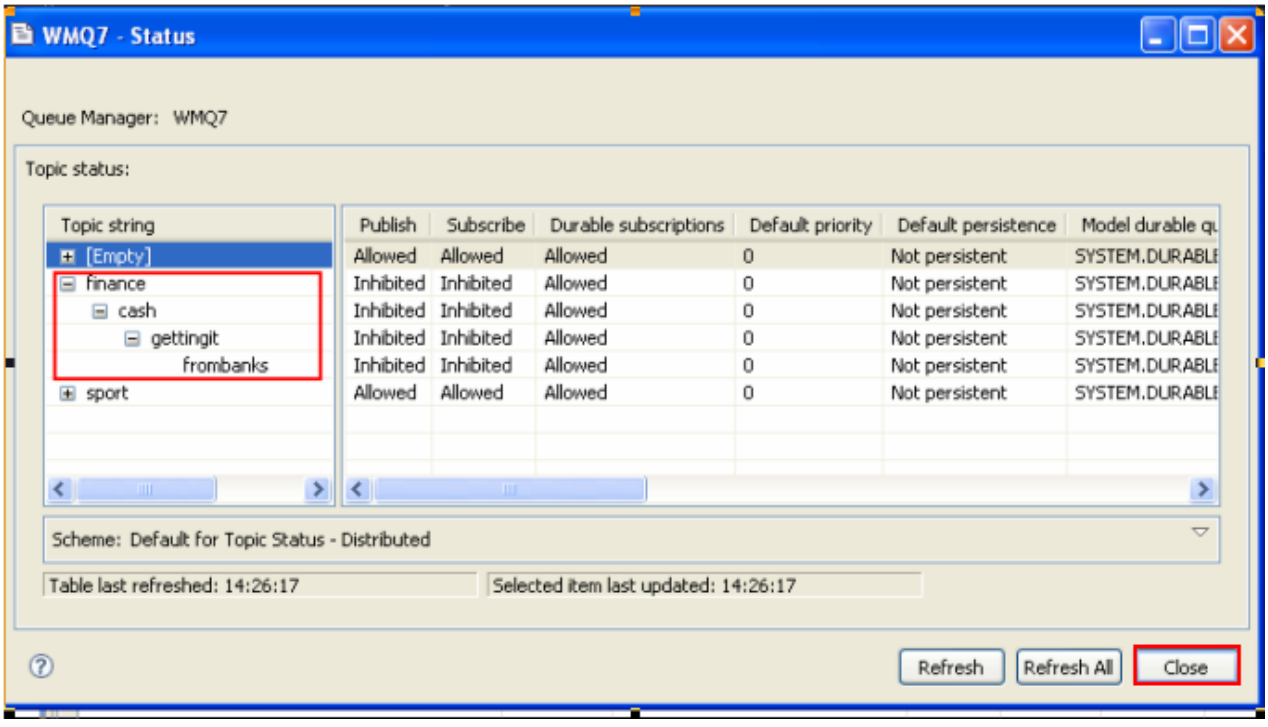


__13. Back in the MQ Explorer, right click on "**Topics**" (under WMQ7) and then select "**Status**".

__14. Now click on the "+" symbol to the left of "**finance"** – a level of the hierarchy opens; repeat on the "+" in front of "**cash**", then "**gettingit**", then "**frombanks**" which is the bottom of the "*tree*".

In the Topic Status notice that all the intermediate nodes have been created and that they have inherited properties from the parent **"finance**". These intermediate nodes have no related Topic Objects and so cannot have their properties altered by MQSC or MQ Explorer.

__15. Close the Topic Status view by clicking on "**Close**".

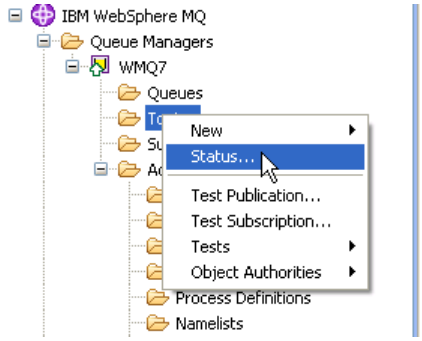


This concludes this portion of Lab 3.

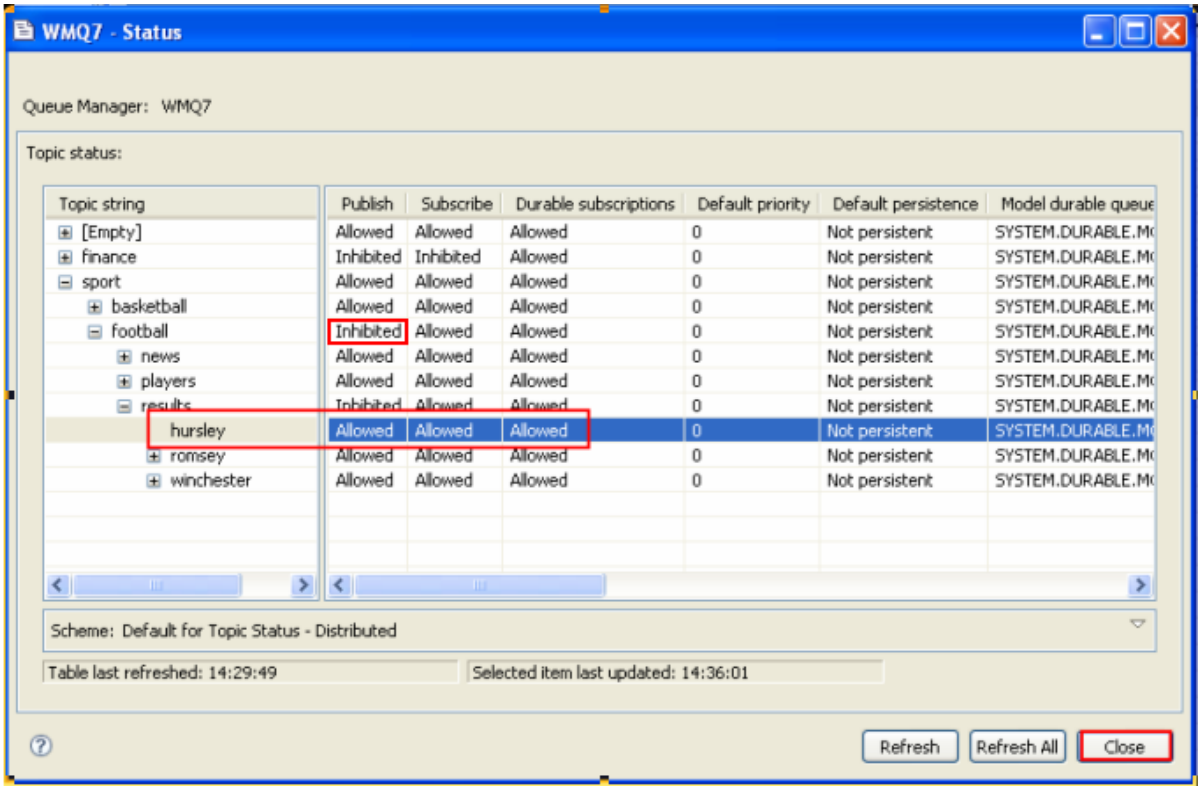
3.3 A First look at the MQ Explorer Pub / Sub Test tools

In this section you will be using the tools that come with WebSphere MQ Explorer that allow you to test publishing to and subscribing to topics.

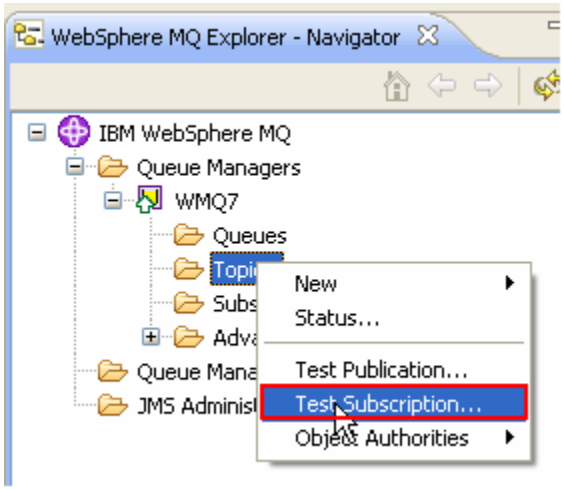
__1. You will now be working with some pre-defined topics. From the WebSphere MQ Explorer, display topic status by right-clicking on the **Topics** folder and selecting **Status…**



__2. You are going to focus on the "**sport**" topic tree. Expand the "+" symbols on the sport topic tree and you should see something similar to the screen capture below. Notice that Publish is allowed for the topic string '**sport/football/results/hursley**'. Also, following the tree "up", you should notice that the topic string **"sport/football"** has the publish attribute **"Inhibited".** Remember this as you complete the next steps of this lab. Click **Close** to dismiss the status window.

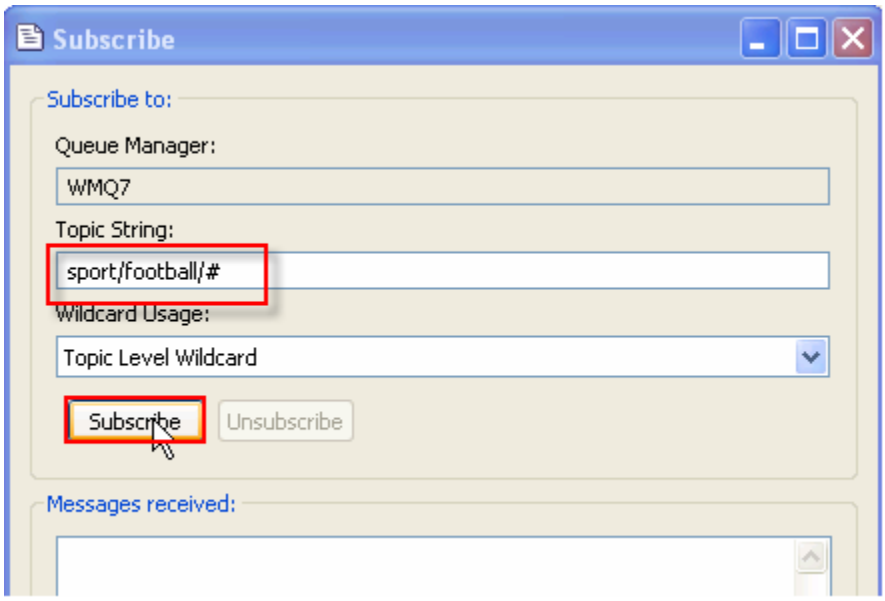


__3. Start a test subscription window by right-clicking on Topics and selecting "**Test Subscription".**



__4. Type in the topic string "**sport/football/#"** and press the "**Subscribe"** button.

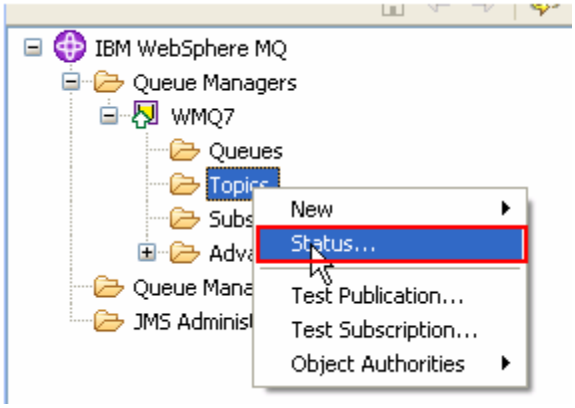
The "#" symbol is called the *multi-level wildcard*. The string **"sport/football/#"** indicates a subscription to all publications sent to the sport/football topic or any of its children. The Test Tool window remains open and the **"Unsubscribe"** button becomes active. Publications received will be displayed in the **"Messages received**" box.



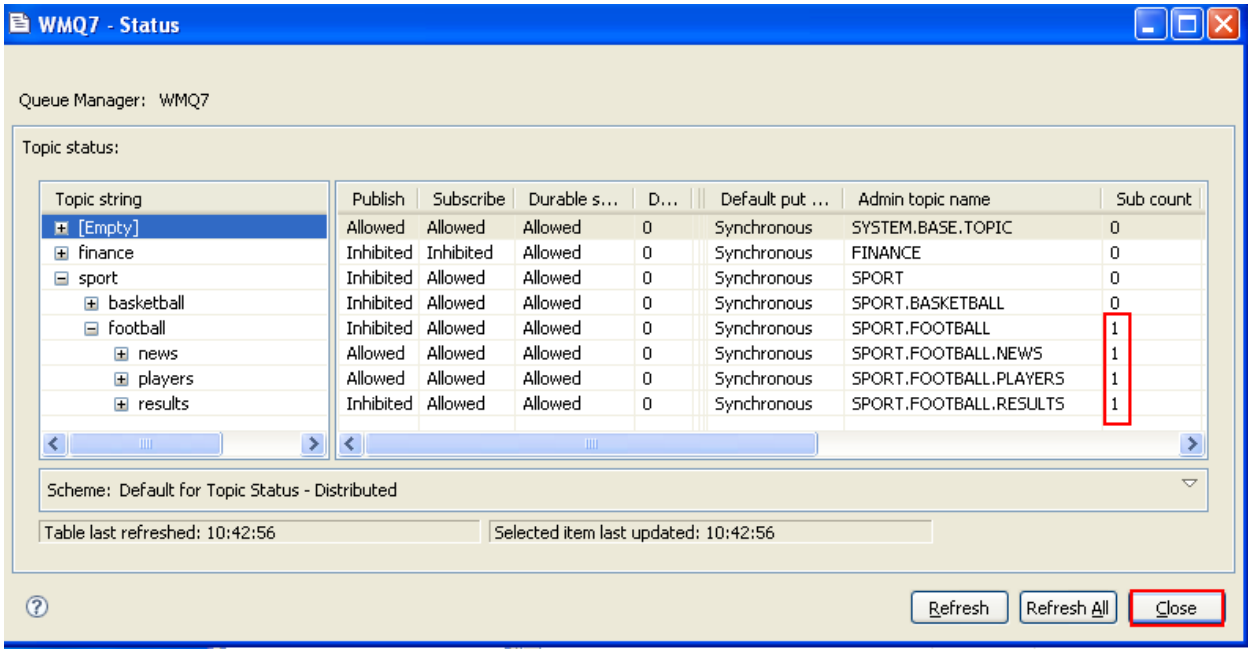
__5. Minimize the Subscribe window by clicking on the minimize button.

__6. The Subscribe window will “park” itself at the bottom left part of the Windows desktop. You will restore this window in a later step of this lab.

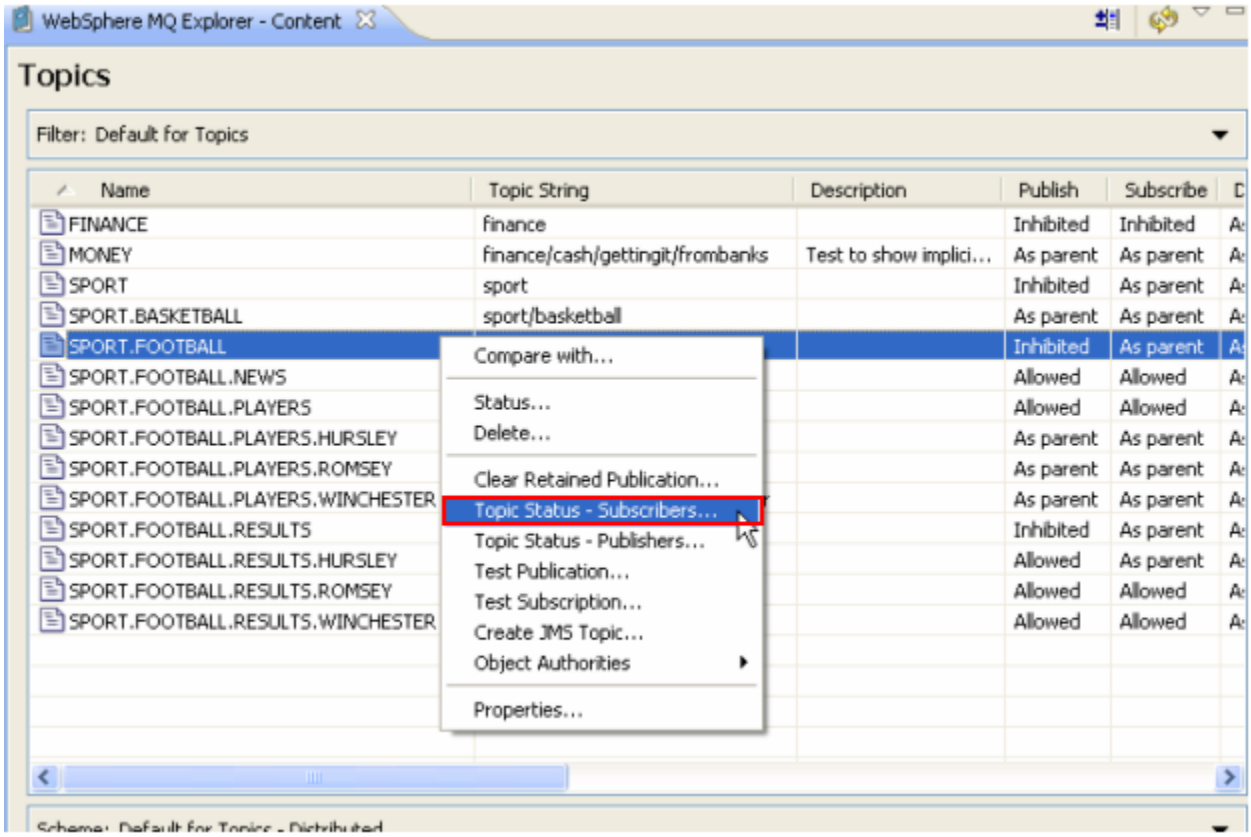
__7. Right click on the Topics folder then select **Status…**



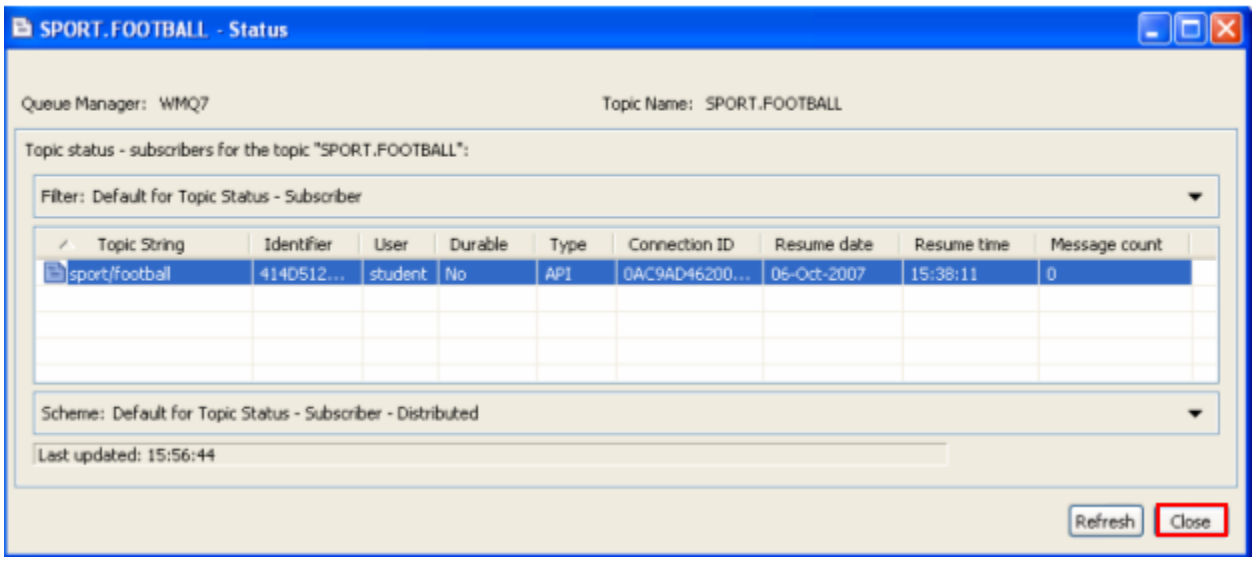
__8. Expand the **sport** tree. Observe the Subscription counts; sport/football and its children have a positive subscription count. Close the topic status window.



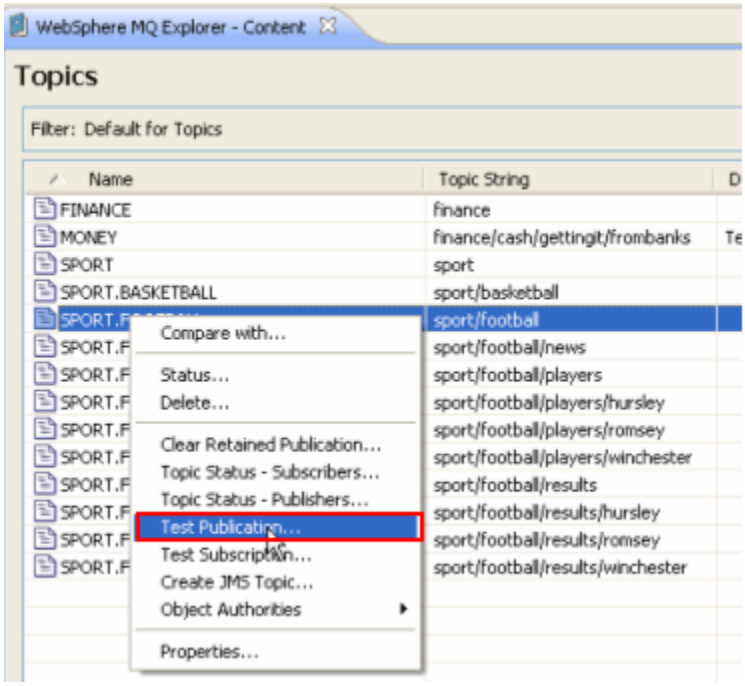
__9. From the Topic display, select the "**SPORT.FOOTBALL**" row, right-click and select "**Topic Status – Subscribers".** This gives detailed information about subscribers to this Topic object.



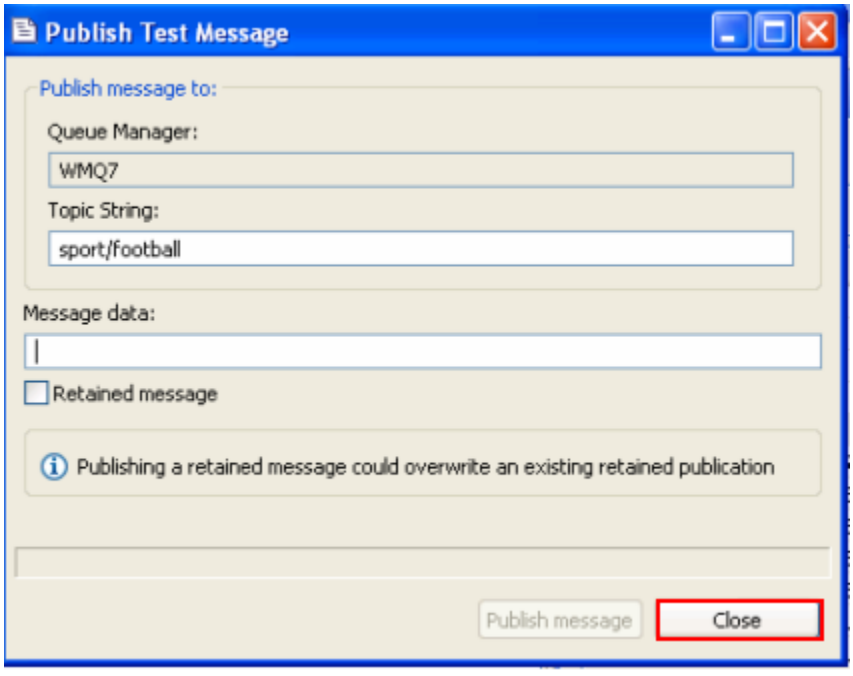
__10. Observe the detailed display and then close the status screen by clicking on the **"Close**" button.



__11. Now you will publish a message. Returning to the Topic list, select the **SPORT.FOOTBALL** row, right-click and then select "**Test Publication**".



__12. This dialog will publish a message to the topic string **"sport/football**".

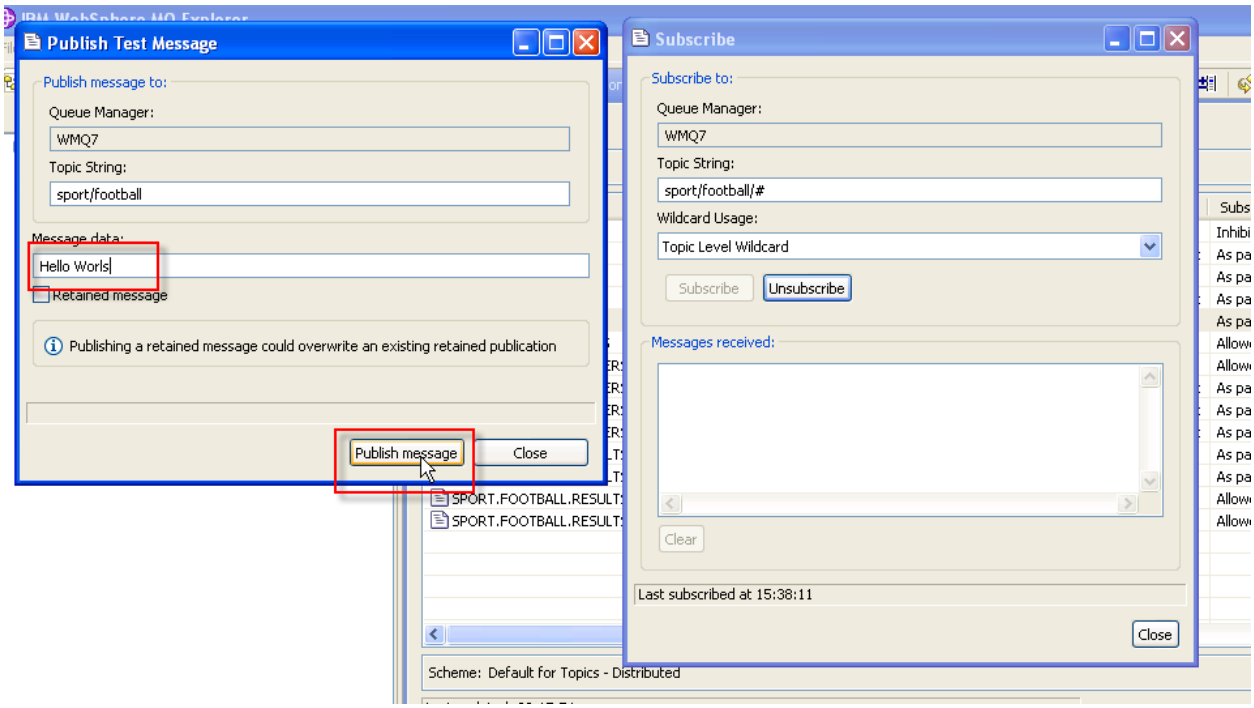


__13. Before entering a message and sending it you will arrange the windows on the screen.

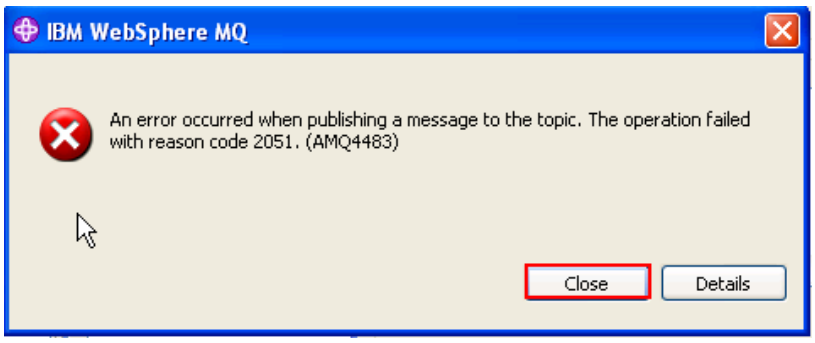
__14. Locate the "**Subscribe**" Test Tool window where you previously subscribed to "**sport/football/#"**. Click on the Restore window button to restore the window. Now position the **Publish Test Message** and the restored **Subscribe** test tool windows so they both are visible. Then return focus to the **Publish** window.

__15. Type a message such as "**Hello World**" and then press "**Publish Message**".

__16. An error occurs because the topic object for "**sport/football**" is **publish-inhibited**; you'll recall that we saw that this was set earlier in the lab. But this will not inhibit our subscribers; we used the multi-level wildcard to subscribe to topics at and below "sport/football" in the topic tree, so we will be subscribing to items published lower in the hierarchy.

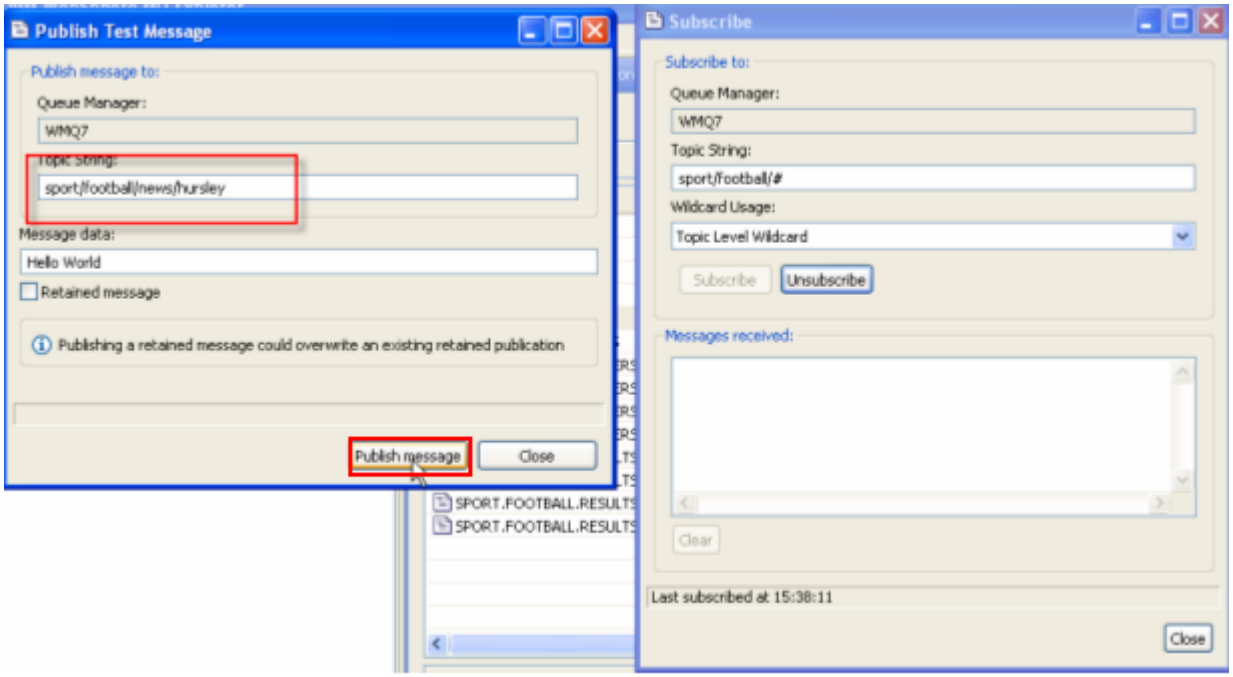


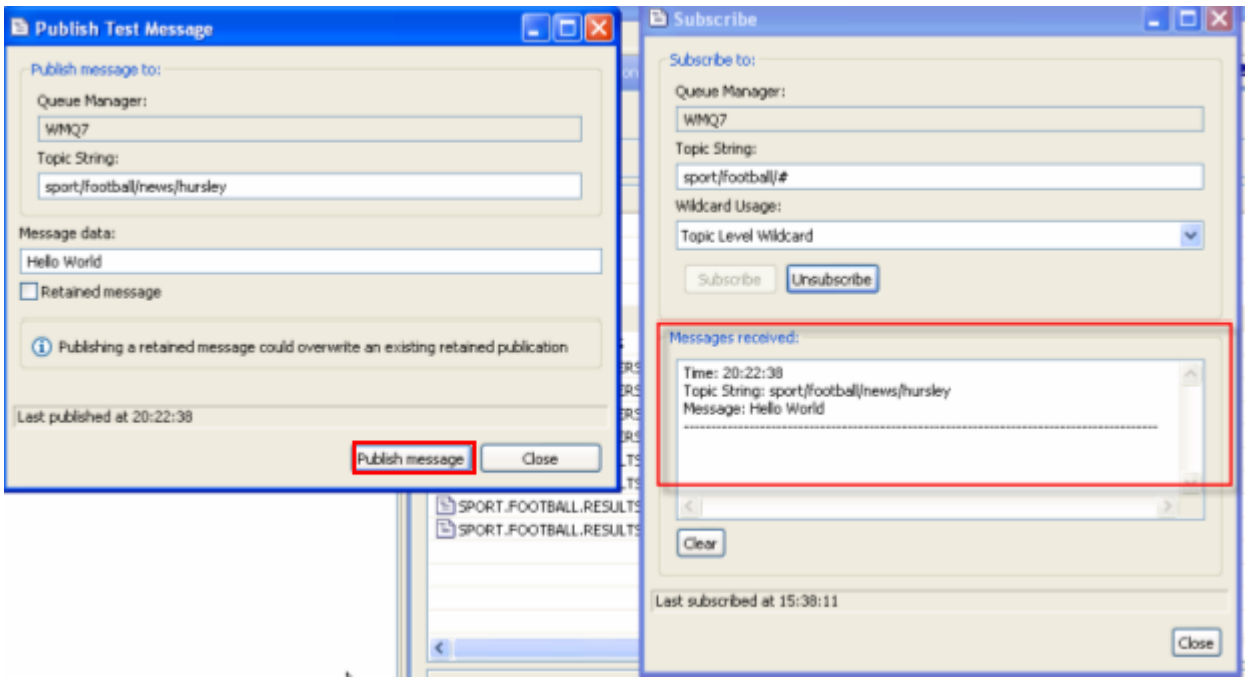
__17. Click on Close to dismiss the error popup.



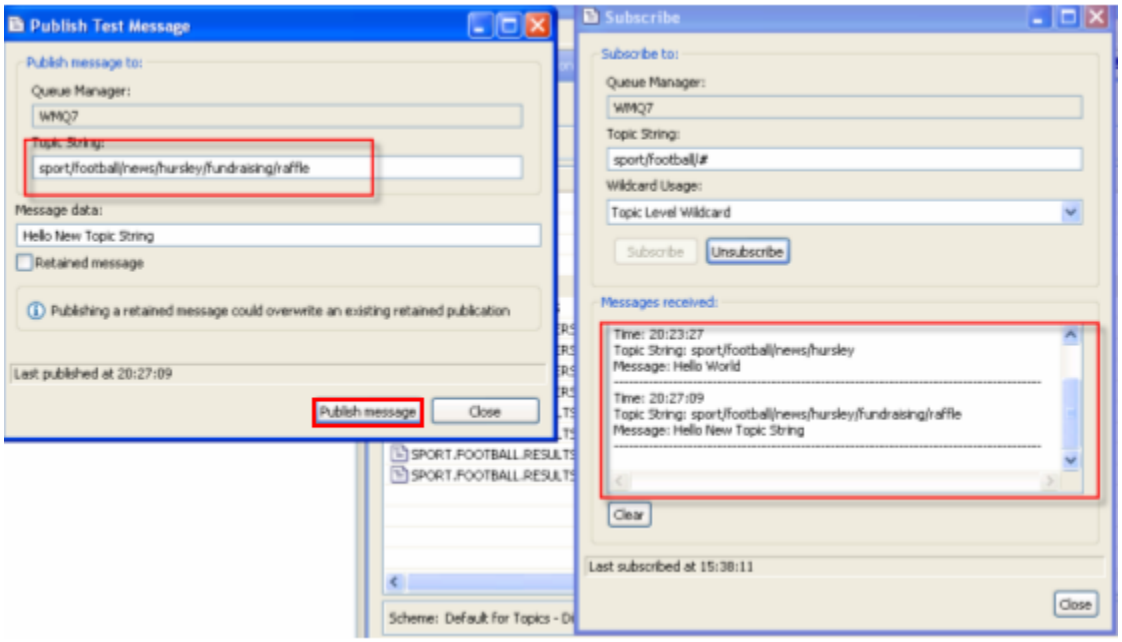
__18. In the Publish Test Message window, overtype the topic string to "**sport/football/news/hursley**" and click on **Publish Message**. You have published and subscribed your first message!

Note: these dynamically created Topic objects are temporary and only exist for a limited amount of time before the queue manager removes them; for example if you restart the Queue Manager, they will no longer exist

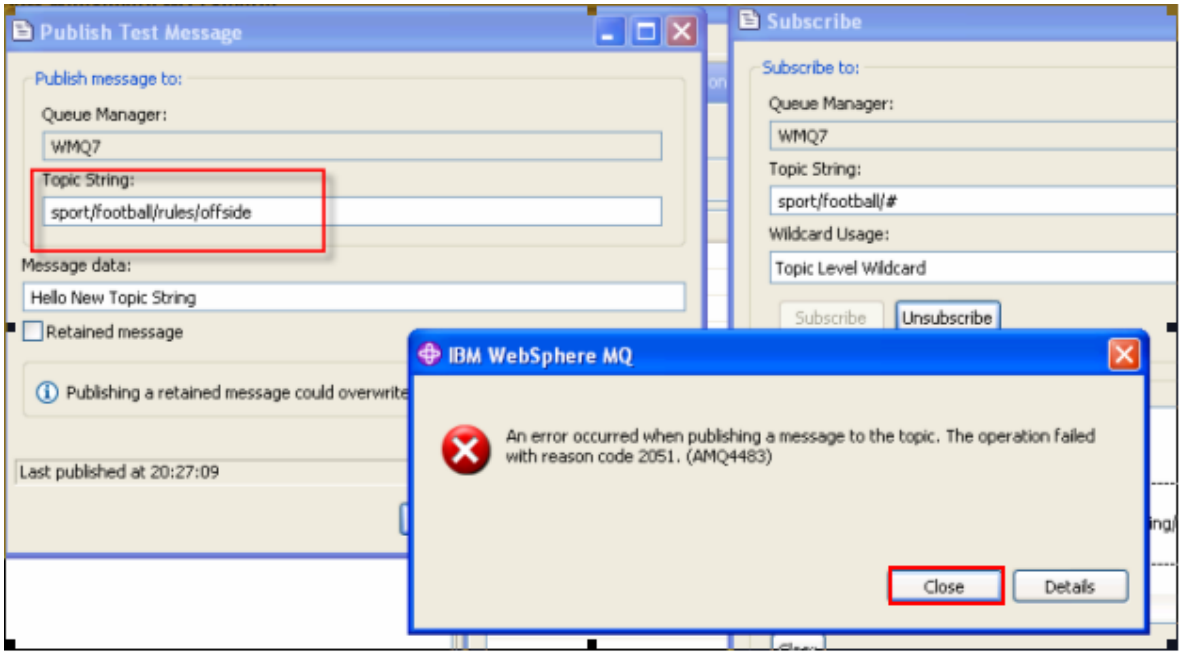




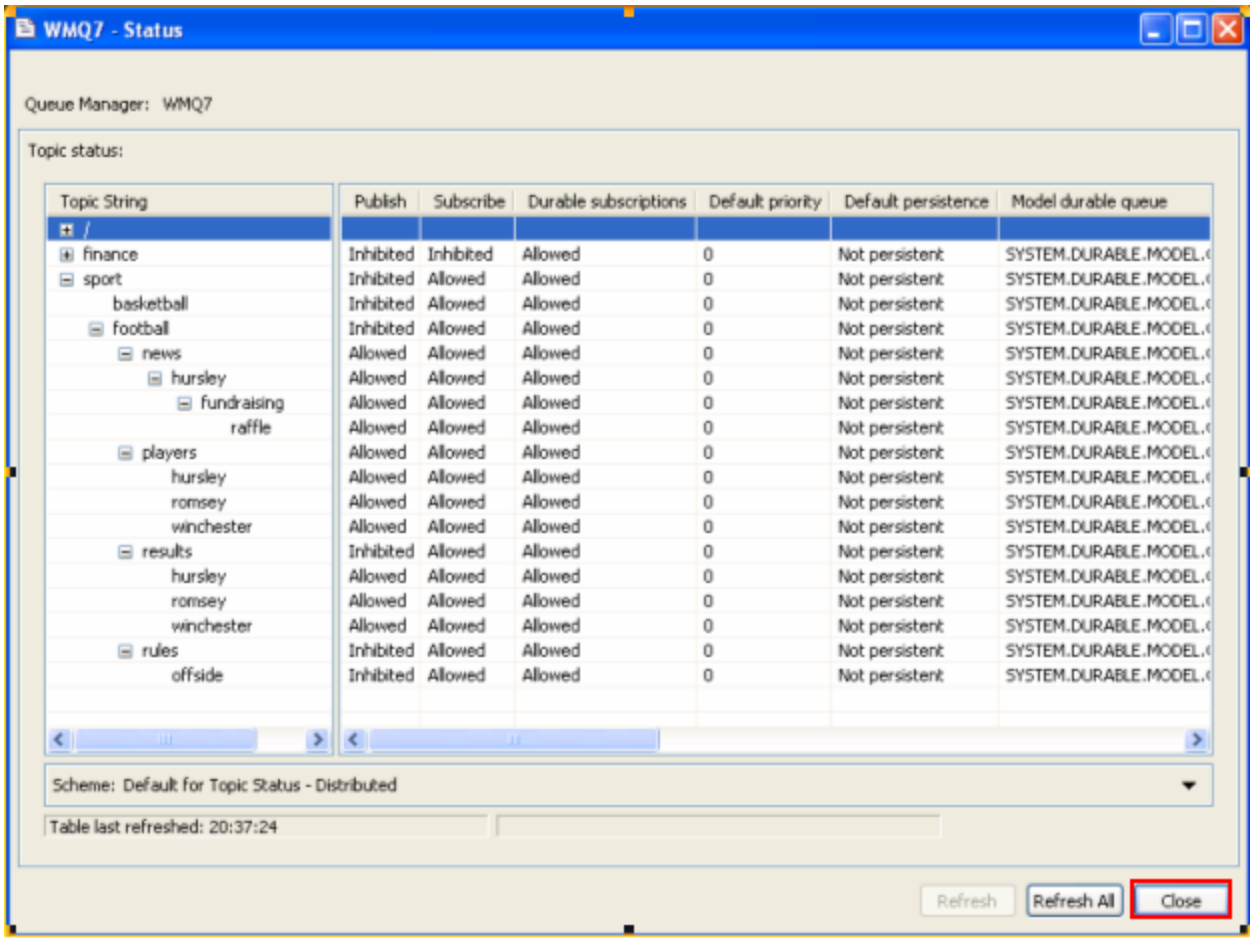
__19. Now try publishing to “sport/football/news/hursley/fundraising/raffle”. The message is sent to the subscriber. New levels of the hierarchy have been created automatically.



__20. Now try publishing to “sport/football/rules/offside”. The publish attempt failed! That is because the node in the topic tree that is dynamically created automatically inherits the properties of the parent “**sport/football**” – which has its Publish attribute Inhibited. Click on **Close** to close the error popup.



__21. Return to the Topic object display. Once again open the Topic status list and expand the **sport** hierarchy. You will see the automatically created elements. Click on **Close** to close the status window.

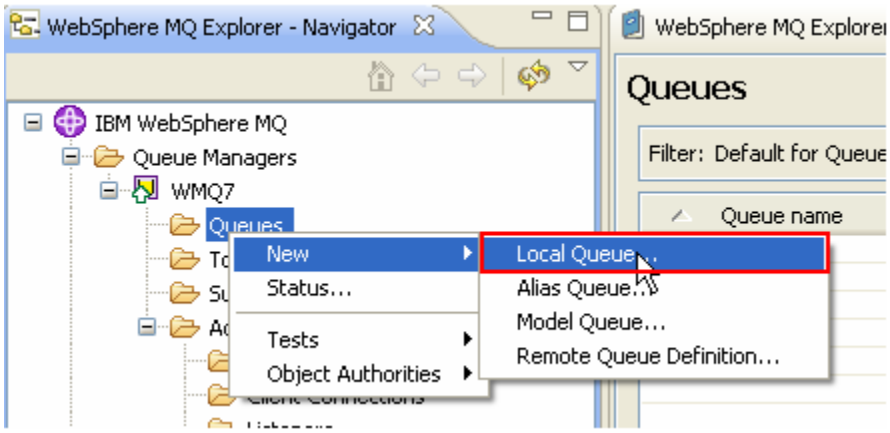


This concludes this portion of Lab 3.

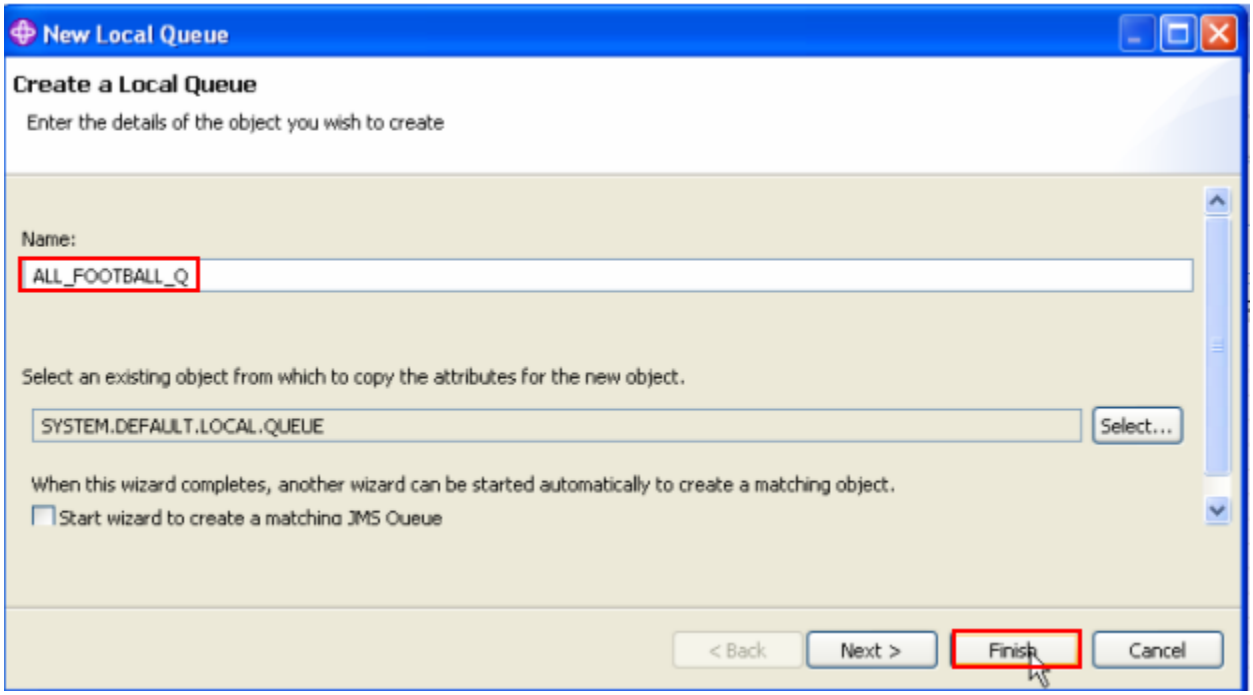
3.4 Administered Subscriptions

While it is typical for subscribers to register their own subscriptions, it is possible to administratively register a subscription using MQ Explorer. This is a subscription to a topic string that delivers messages to a queue. This can be very useful because it is a way for a legacy program which was designed as a point-to-point application to read a queue associated with a topic; in this way it can participate in publish/subscribe without changing the program. You will now explore how such a subscription can be set up and used.

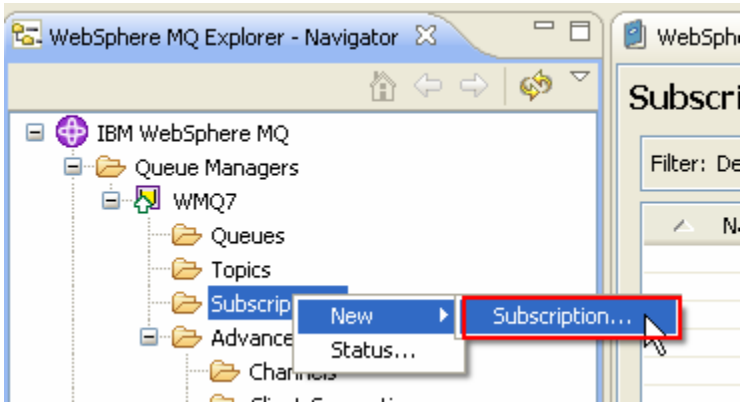
__1. In the MQ Explorer, select “**Queues**”. Right-click and select “**New→Local Queue**”.



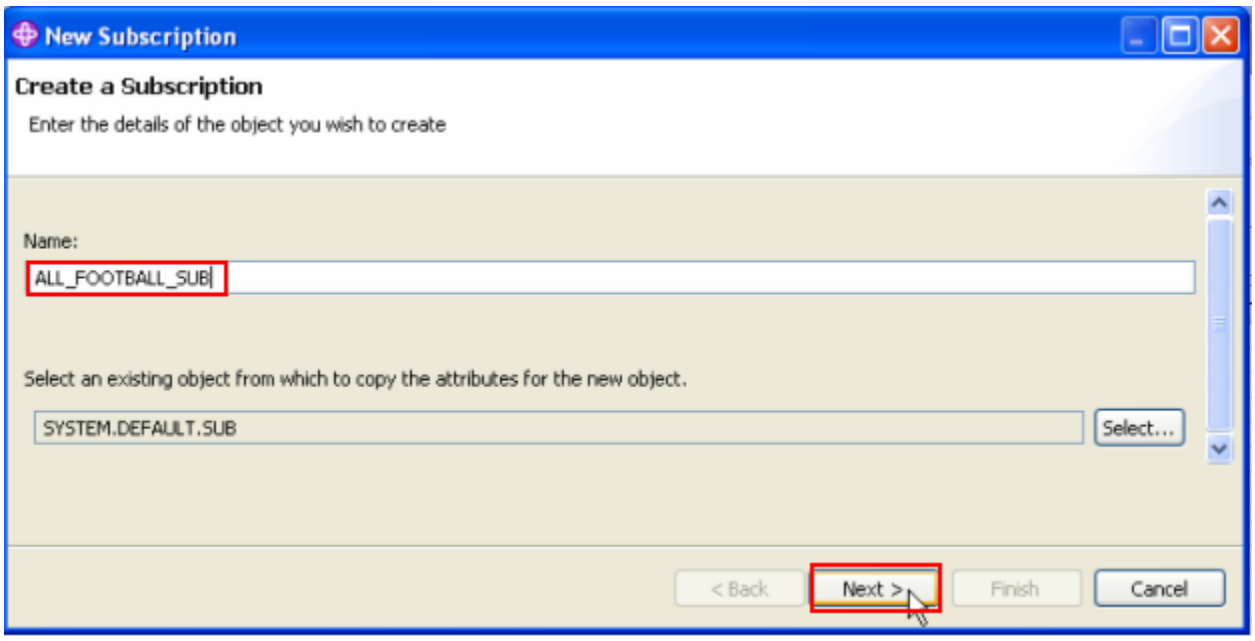
__2. Name the queue “**ALL_FOOTBALL_Q”** and press **“Finish**”.



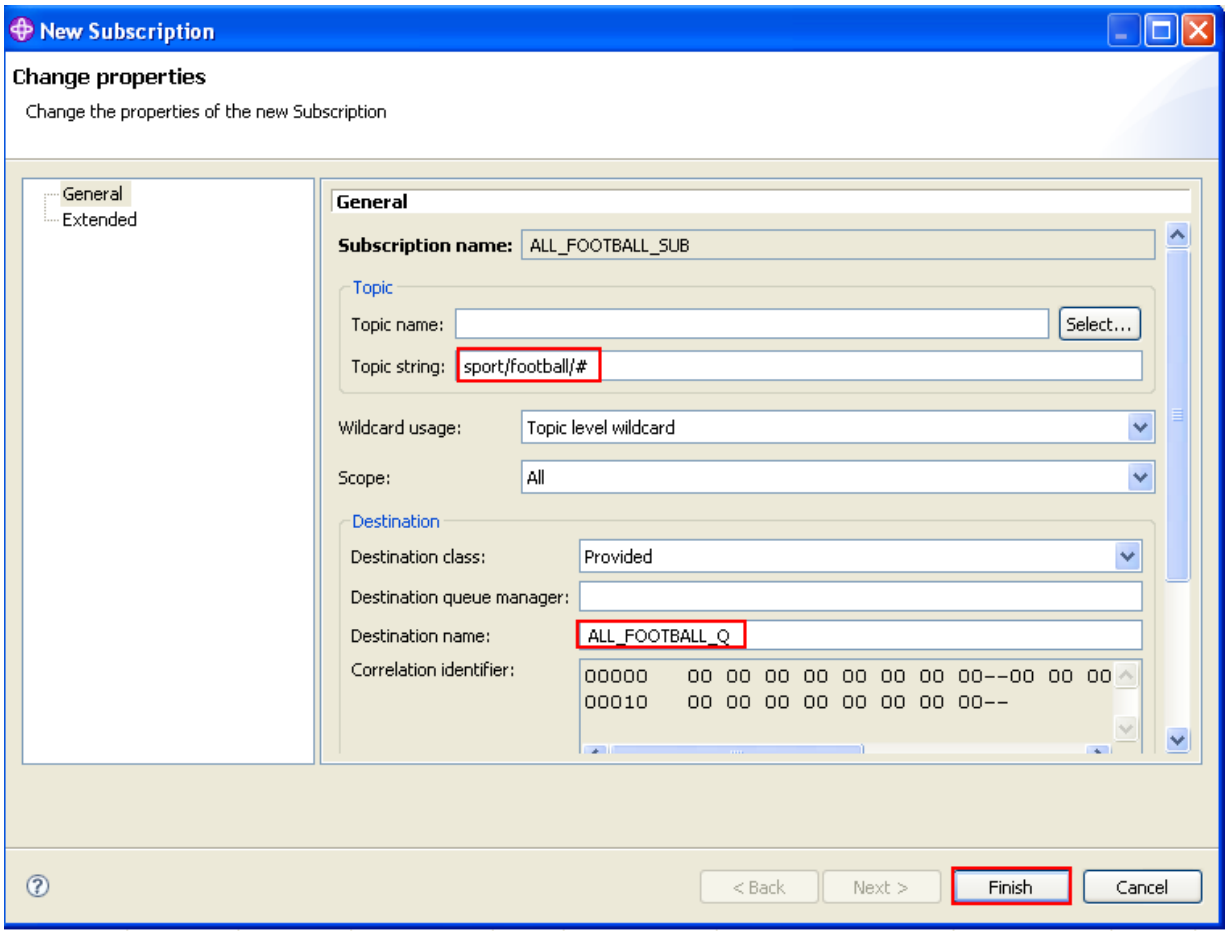
__3. Select “**Subscriptions**”, right-click and select “**New→Subscription**”.



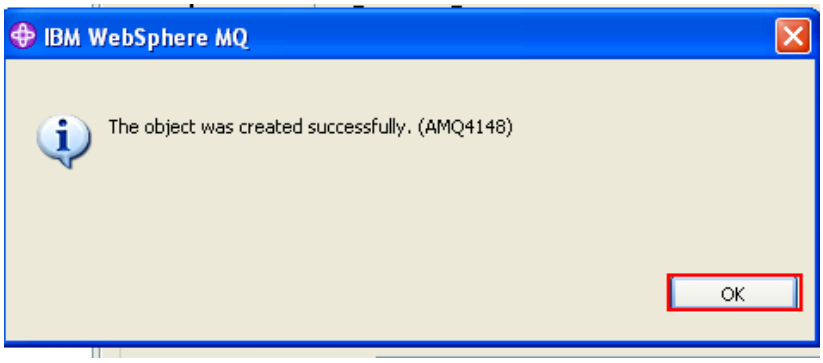
__4. Type **ALL_FOOTBALL_SUB** as the subscription name and then click **“Next**”.



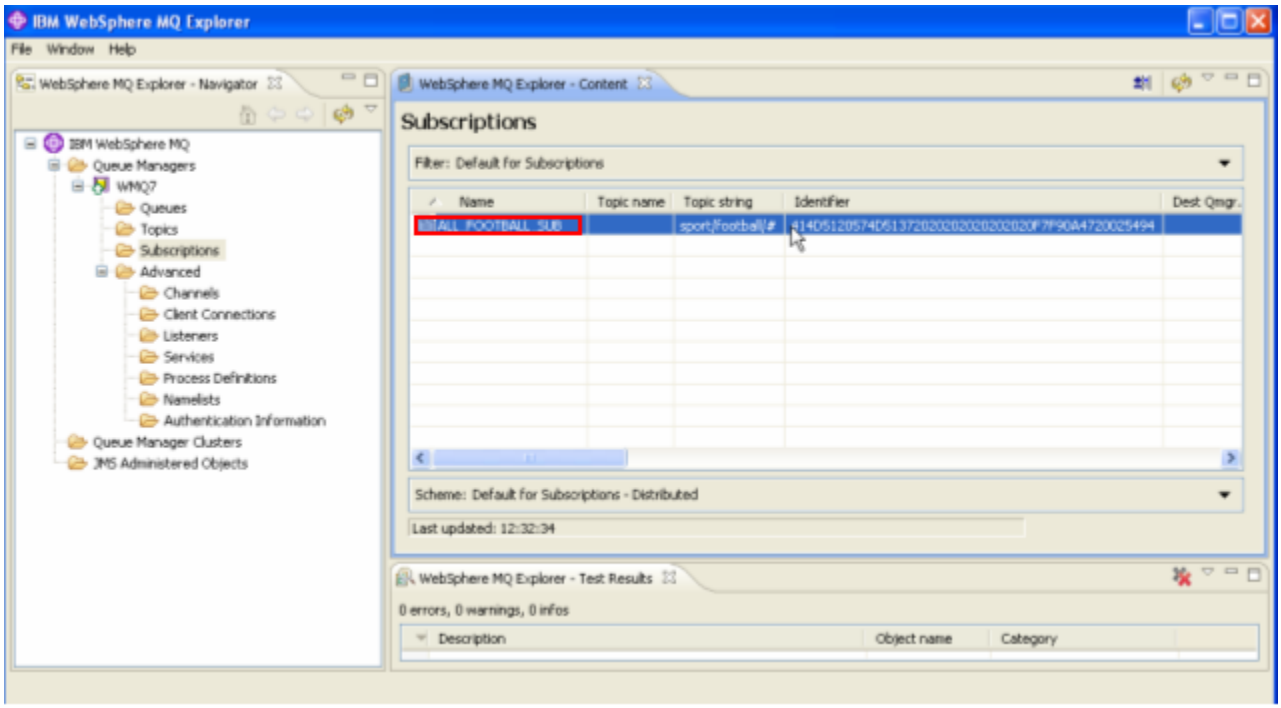
__5. Leave the Topic Name blank, and enter **sport/football/#** as the Topic string. Leave the Destination Queue Manager blank and enter **ALL_FOOTBALL_Q** in the Destination Name. Then click **Finish**



__6. Click **OK** to close the confirmation window.

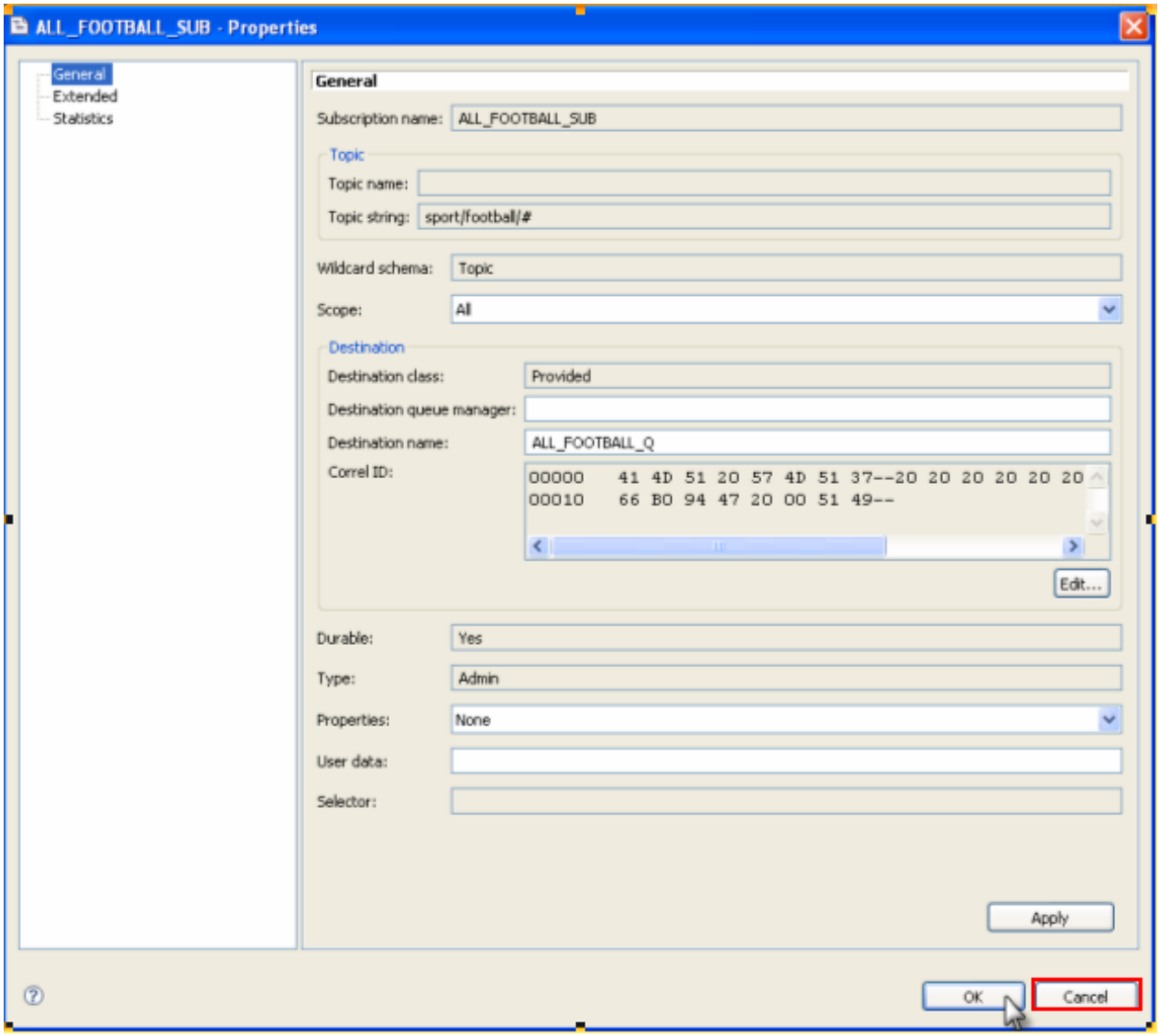


__7. The new administrative subscription appears.



Double-click on the new subscription to see its attributes.

__8. The attributes of the new subscription are displayed.



This subscription will now route all qualifying messages to the local queue **"ALL_FOOTBALL_Q"**

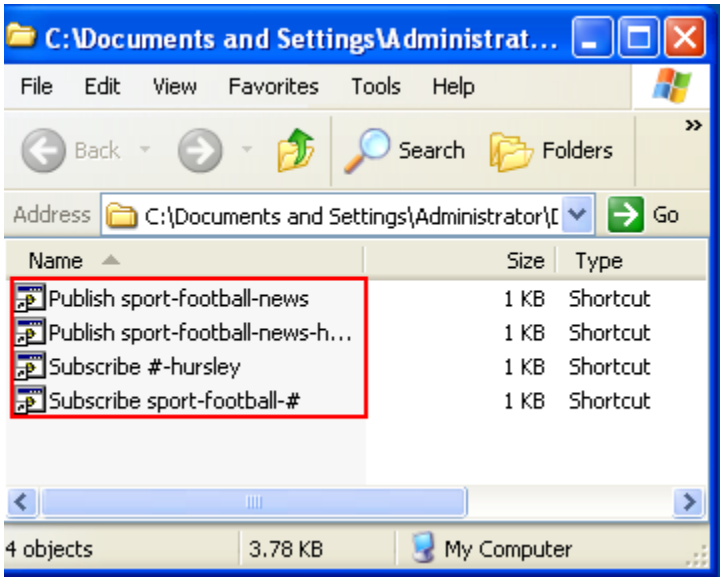
__9. Close the properties window by clicking on **Cancel.**

This concludes this portion of Lab 3.

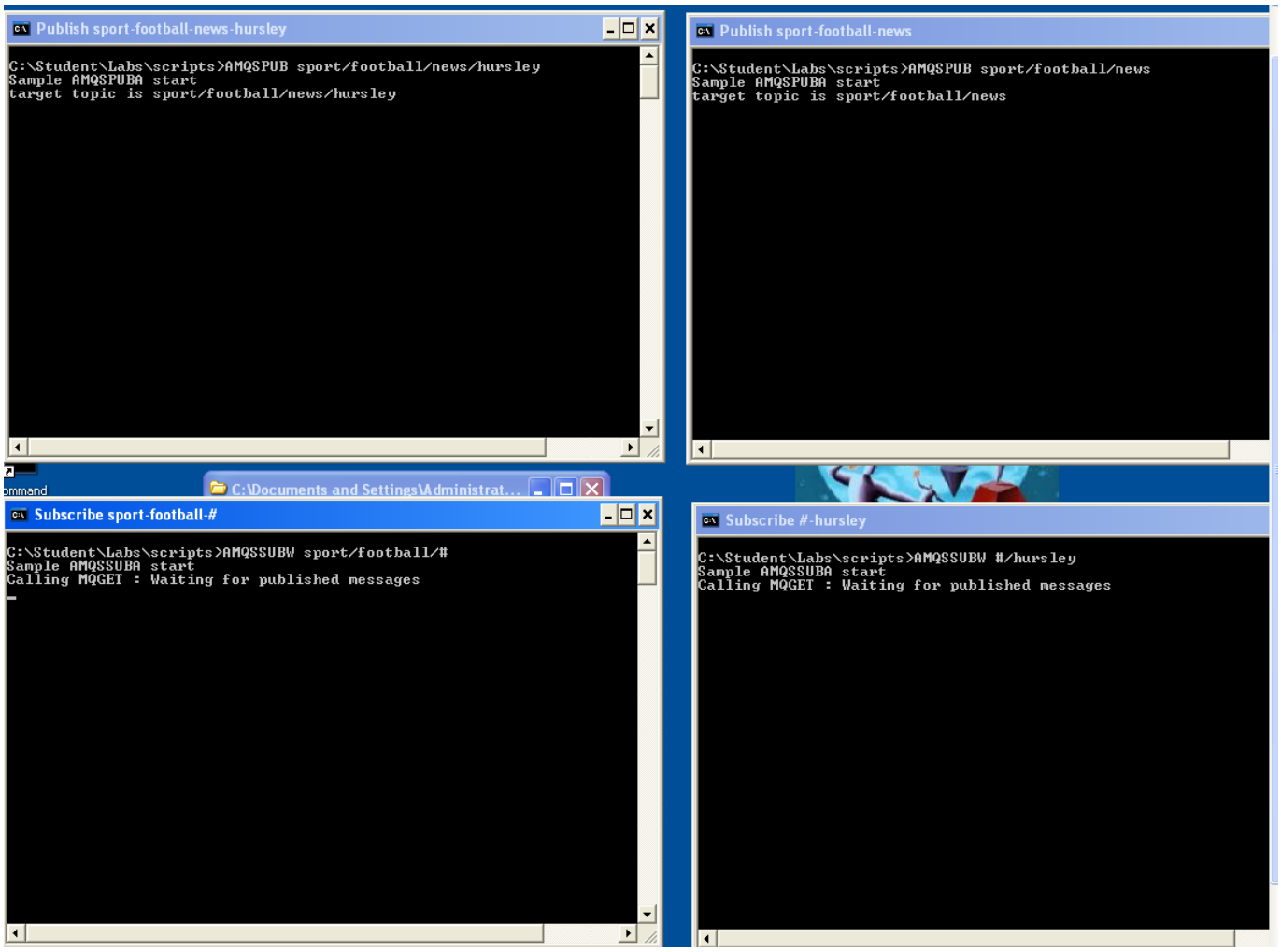
3.5 Testing Publications and Subscriptions from the command line

You will now use two more sample programs that are supplied with WebSphere MQ to further test WebSphere MQ publish and subscribe capabilities, called amqspub and amqssub.

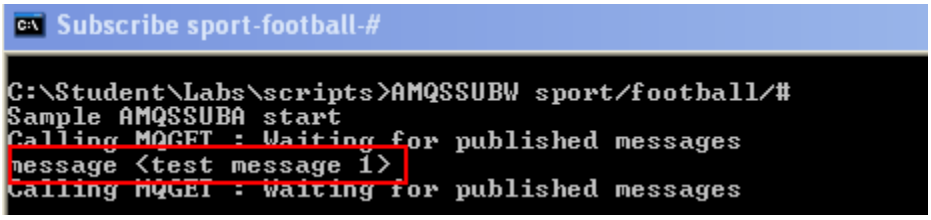
__1. A folder on the desktop contains four shortcuts that will start two instances of a publishing sample, and two subscribers. Open the folder and then double-click each shortcut to launch the programs.

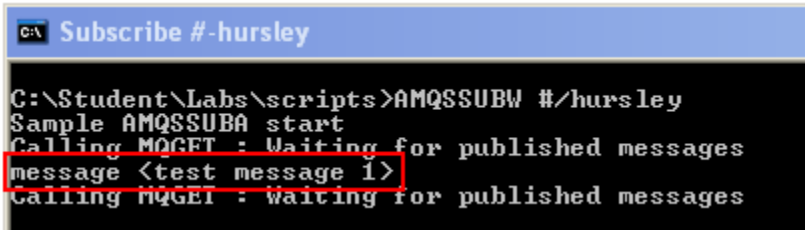


__2. The top two windows are the *topic publishers* (amqspub) Each time you type text into either window, the windows on the bottom, the *topic subscribers* (amqssub) will receive the text as published messages because the topic string that they are subscribing to matched the one being used by the publishers.



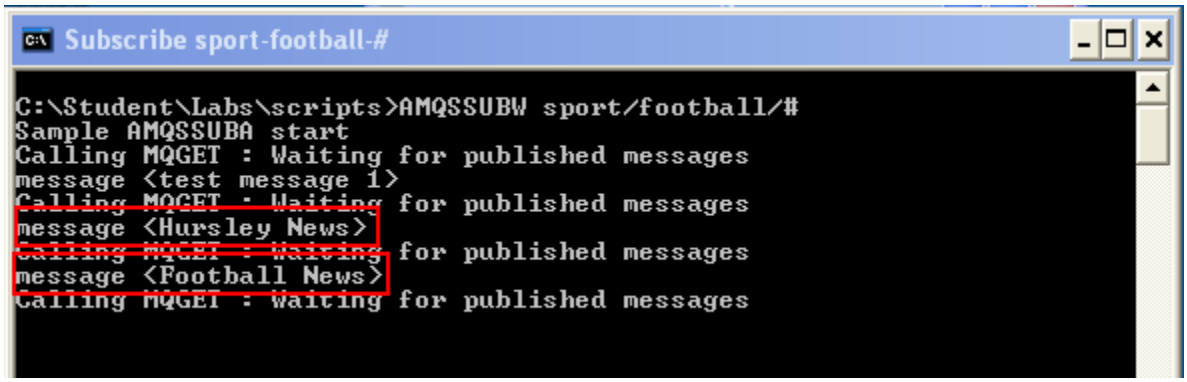
__3. Now in the top left window (publishing to **sport/football/news/hursley**) enter "**test message 1**" and press Enter. The message should appear in *both* subscribing windows because the published message matched both subscriptions.



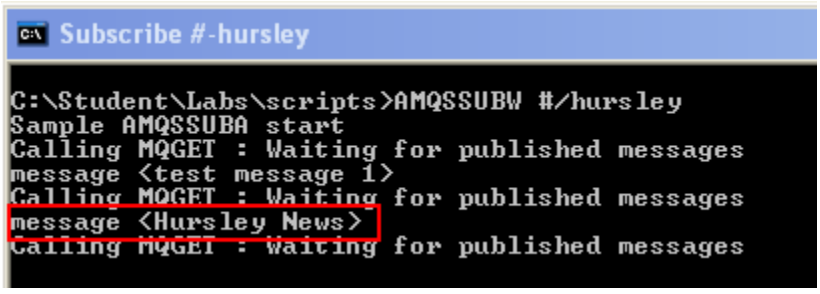


__4. Again in the top left window type the text "**Hursley News**" and then press "Enter".

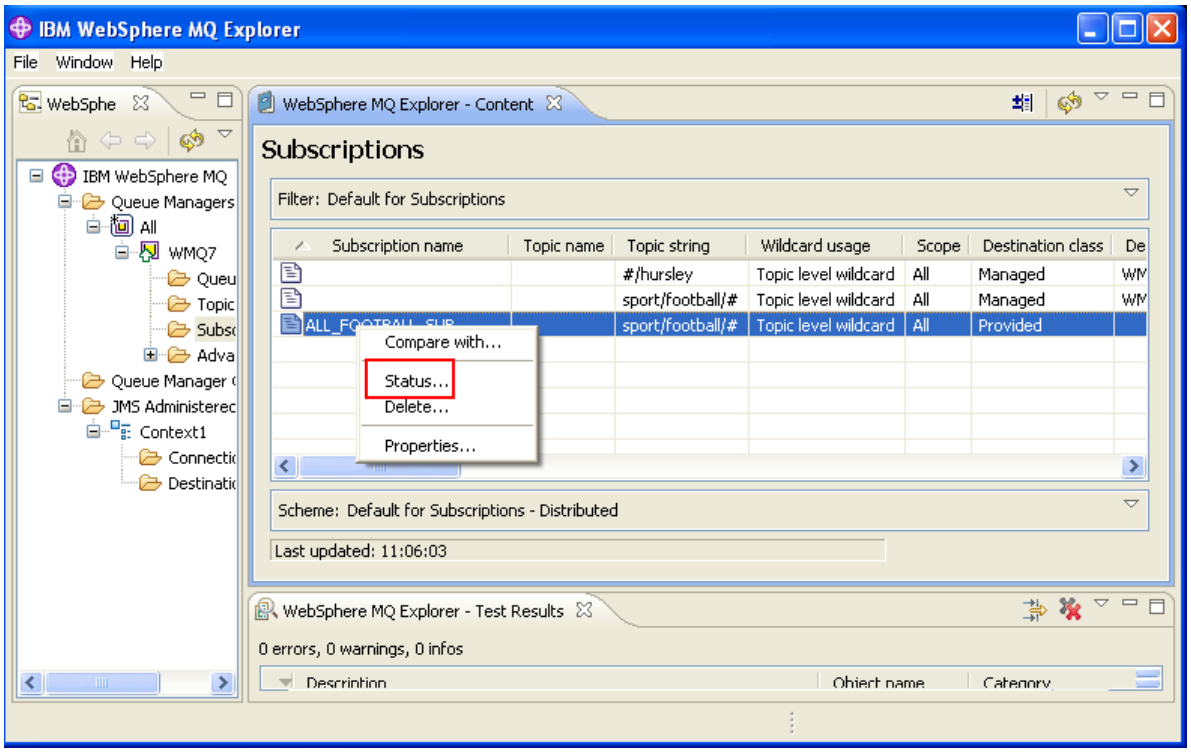
__5. In the top right window type the text "**Football News**" and press "Enter". Notice that the **sport/football/#** subscription gets both publications. This is because when you subscribed you used a *multi-level wildcard* (#) to indicate that you were interested in messages published to the **sport/football**" topic or any of its children, so you will get both messages.



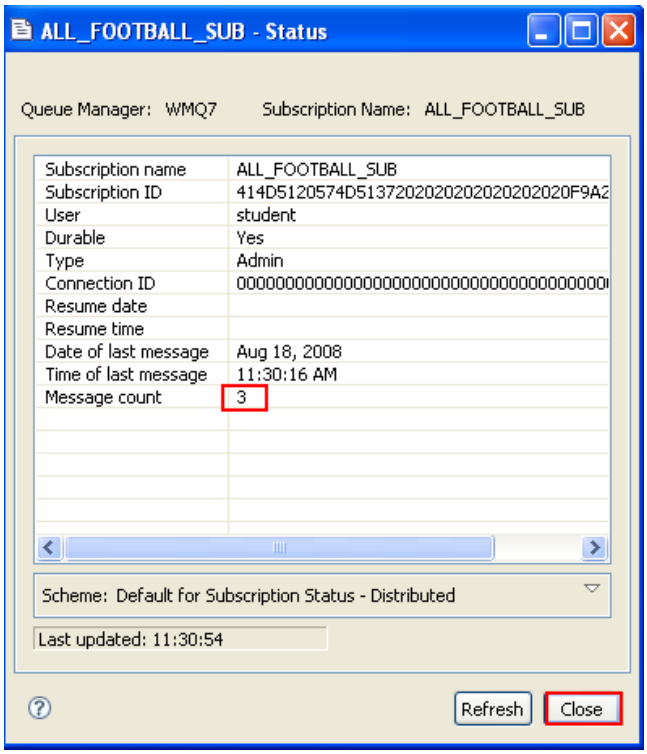
__6. On the other hand, the **#/hursley** subscription gets only one.



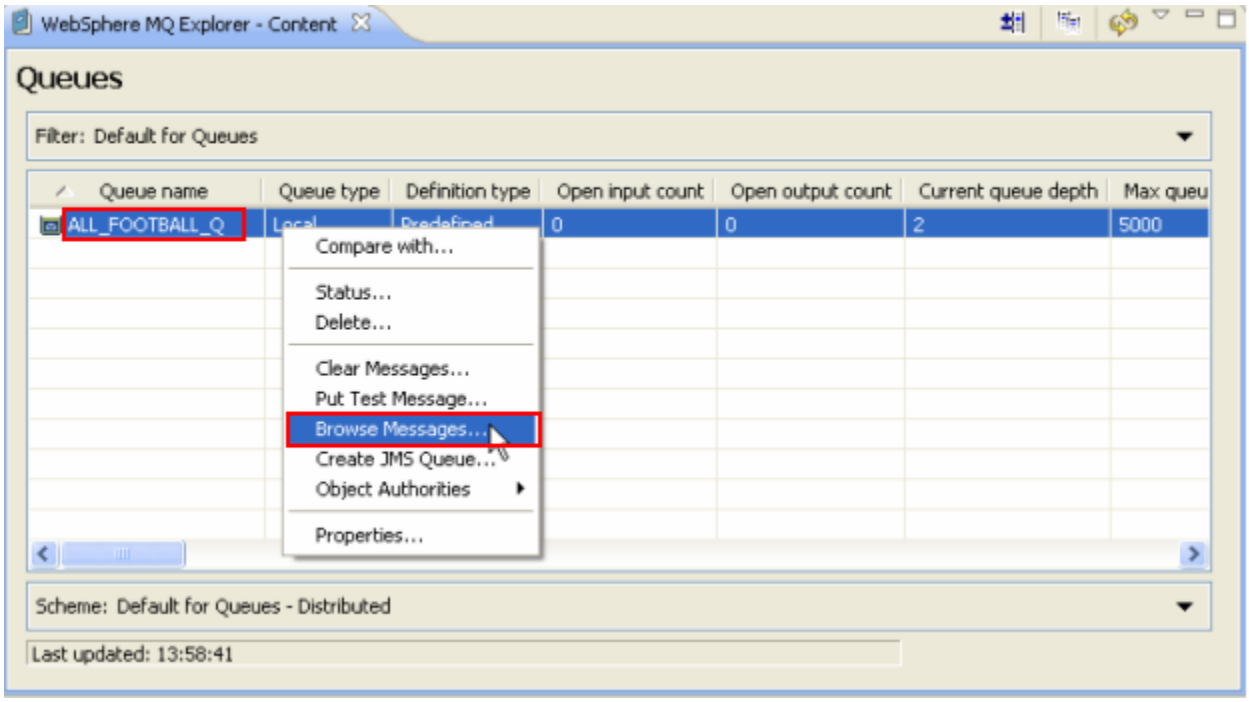
__7. Return to the Subscriptions view in the MQ Explorer. Select the **ALL_FOOTBALL_SUB** subscription, right-click and select "**Status"**.



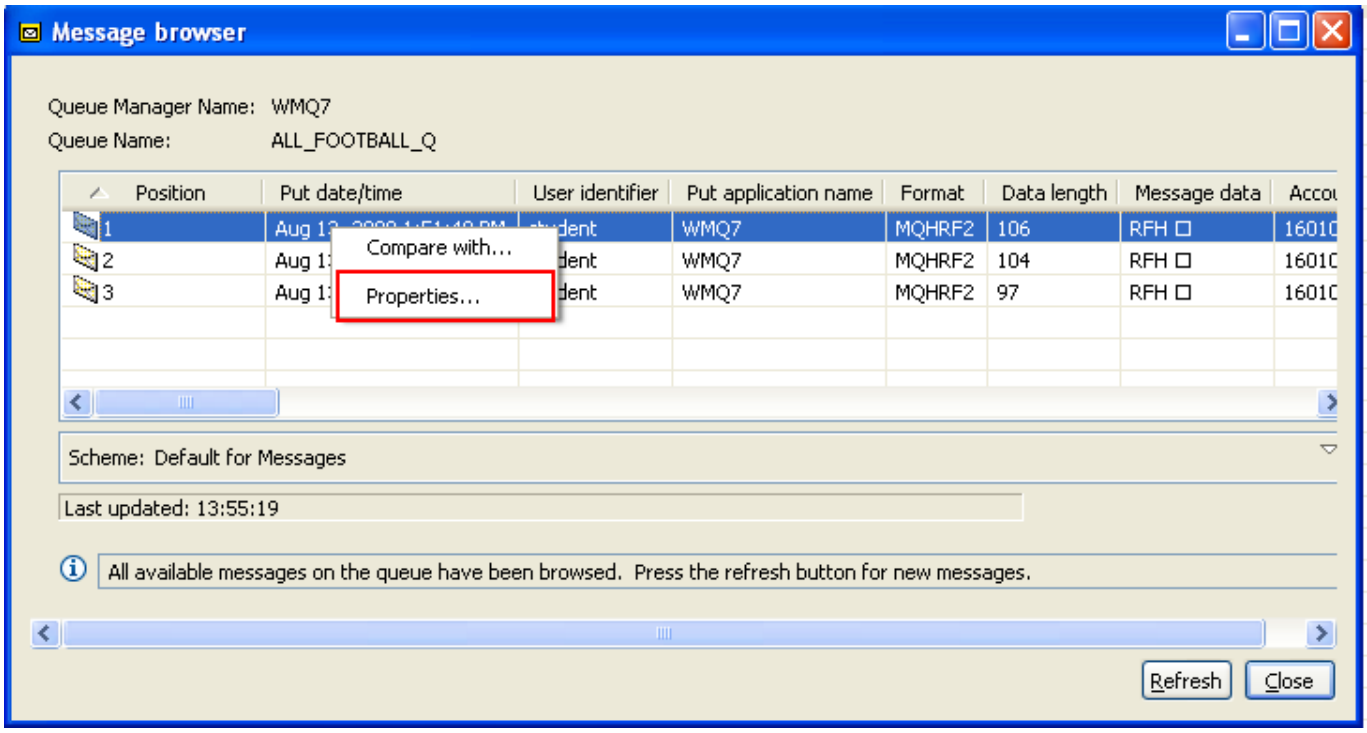
__8. The message count should have a count of the messages that were published on this topic. Click on **Close** to close the status window.



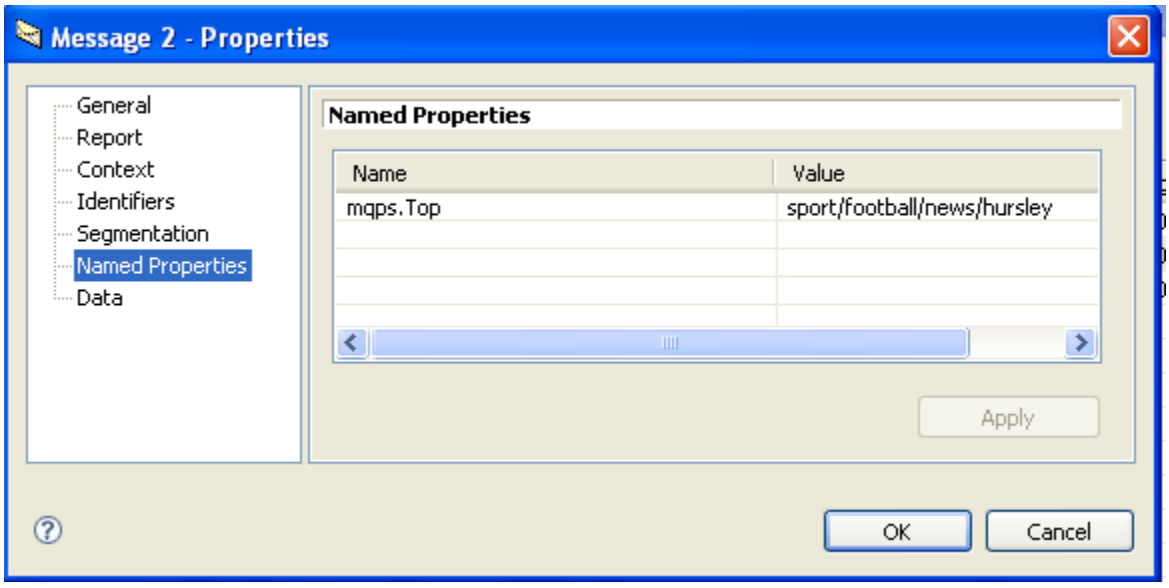
__9. Right-click on the MQ queue **ALL_FOOTBALL_Q** and select **Browse Messages**



__10. You should see three messages on the queue (or as many as you put in the amqspub test). Select one of the messages, right-click and choose "**Properties**".



__11. Click on the “**Named Properties**” tab. From this display you can see the originating topic string.



__12. Close the four or five open command windows as you will no longer need them.

This concludes Lab 3.

Lab 4 WebSphere MQ Security lab

4.1 WebSphere MQ Security Lab Overview

This lab will demonstrate WebSphere MQ security administration for point-to-point as well as publish/subscribe MQ objects using the WebSphere MQ Explorer.

4.2 Review the security on the system

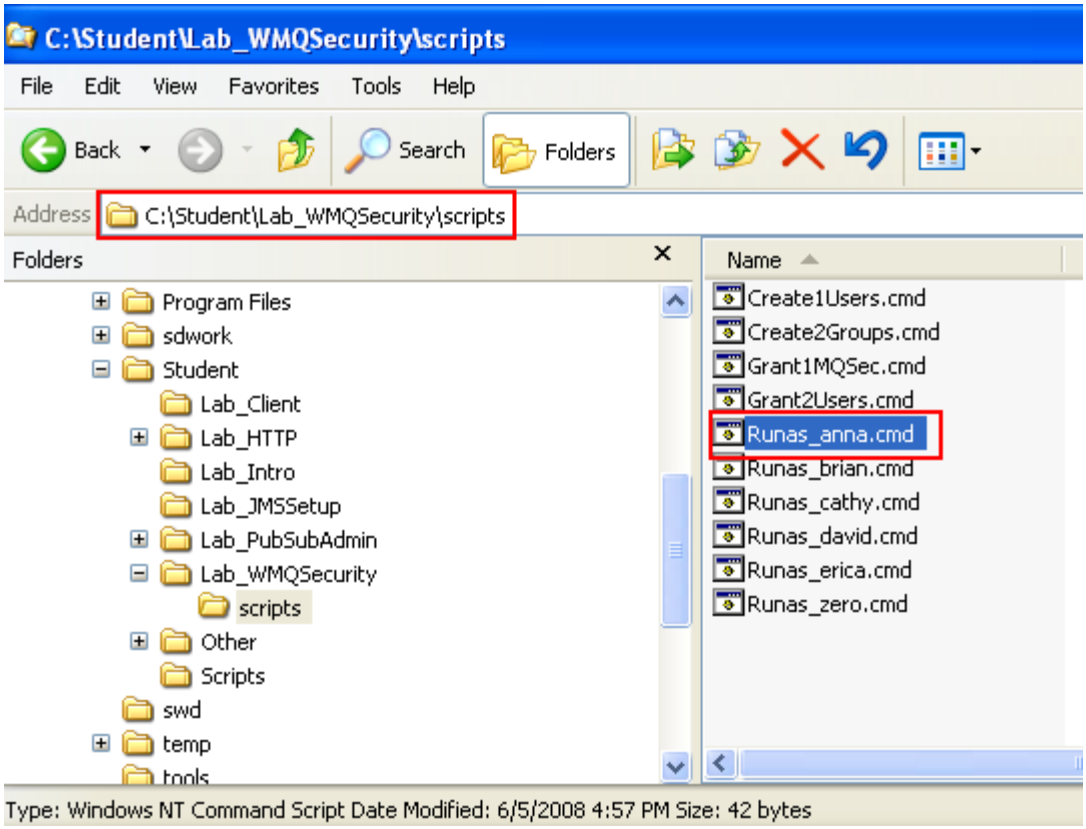
__1. There are several user ids that are pre-defined on your system. You will be using the user called "**Anna**" and also **"Brian"** to demonstrate some of the security capabilities of WebSphere MQ V7.

Username	**Role**
Anna	Superuser, member of the mqm group
Brian	Subscriber
Cathy	Result publisher
David	Journalist, can publish to news
Erica	Manager of Hursley football club
Zero	A user without access to the MQ system

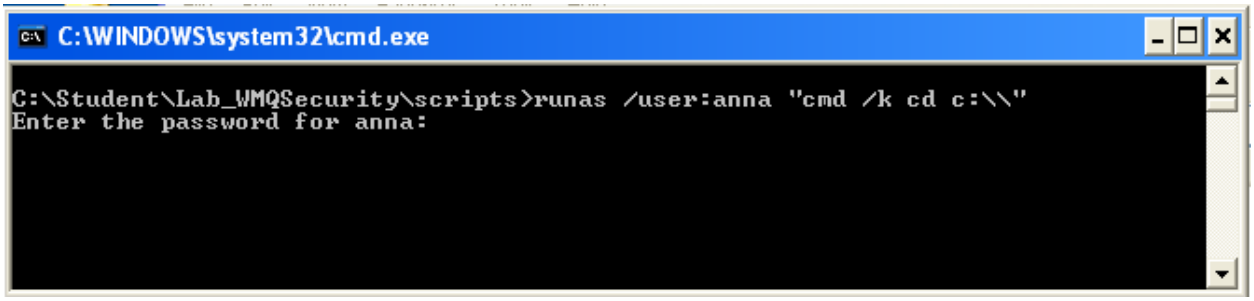
4.3 Using RUNAS to change runtime authority.

__1. Using the Windows explorer, navigate to the directory **C:\Student\Lab_WMQSecurity\scripts**

__2. To run as a different user you will use a program called "**runas**". For example, to **"runas"** user Anna, double-click the script "**Runas_anna.cmd".** This will prompt for your password (which is **passw0rd** with a zero ("0") for the "o") and open a command window running with that user's authority.

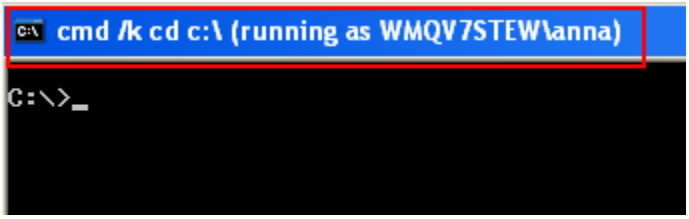


__**3.** Type in the password for Anna which is **passw0rd (**with a zero ("0") for the "o")



__4. You are now running as user Anna.

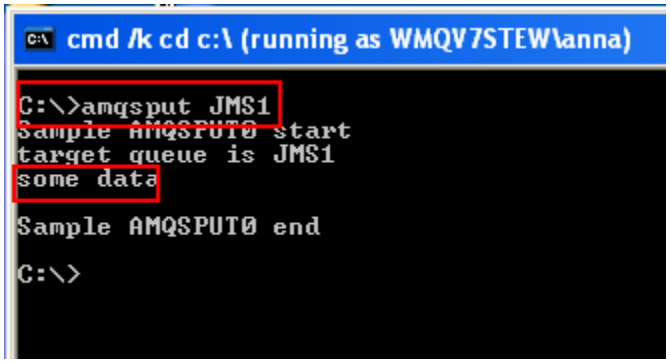
This concludes this portion of Lab 4.



4.4 Point to Point (Queue) Security

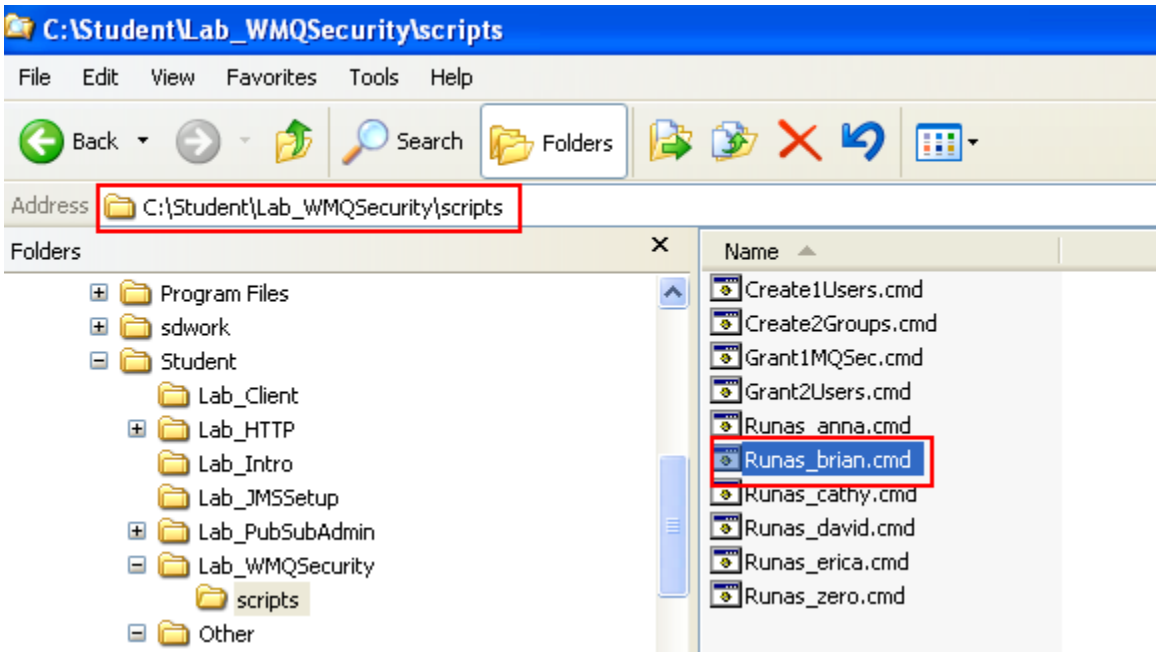
__1. Anna is a “**super user**” because her user id is a member of the **mqm** group. She is able to basically do anything without security stopping her.

__2. In the runas window of Anna, type in the following command: **amqsput JMS1.** Type in some text, press Enter, and then press the Enter key a second time to end amqsput. Everything should work fine as expected.

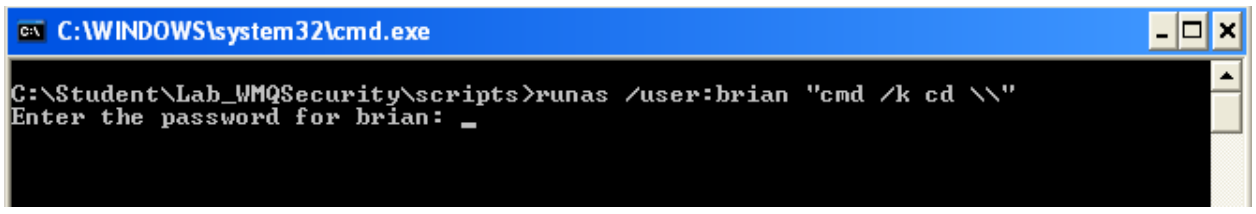


__3. Leave the runas Anna window open, you will need it in a later step.

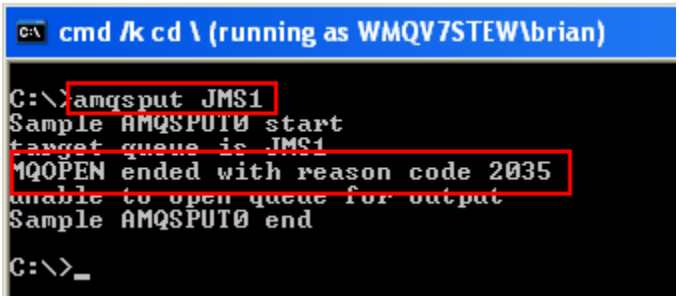
__4. Find the **Runas_brian.cmd** command and double-click to run it.



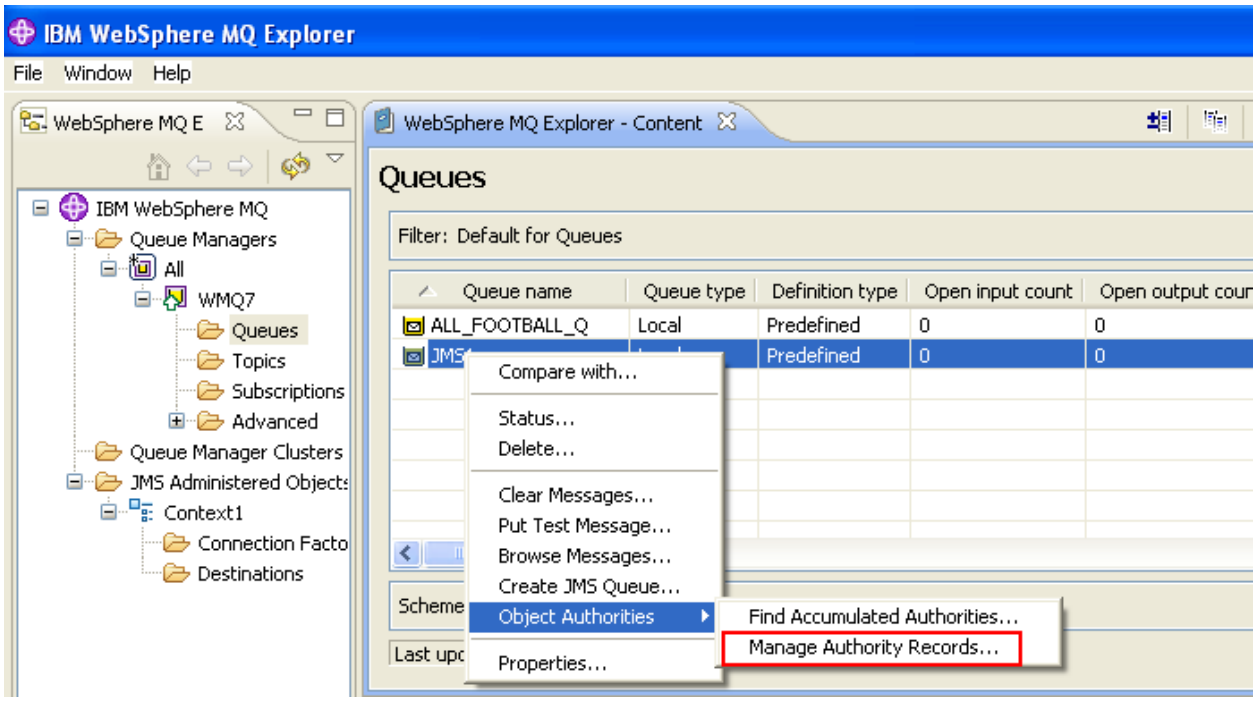
__5. The password is **‘passw0rd.**



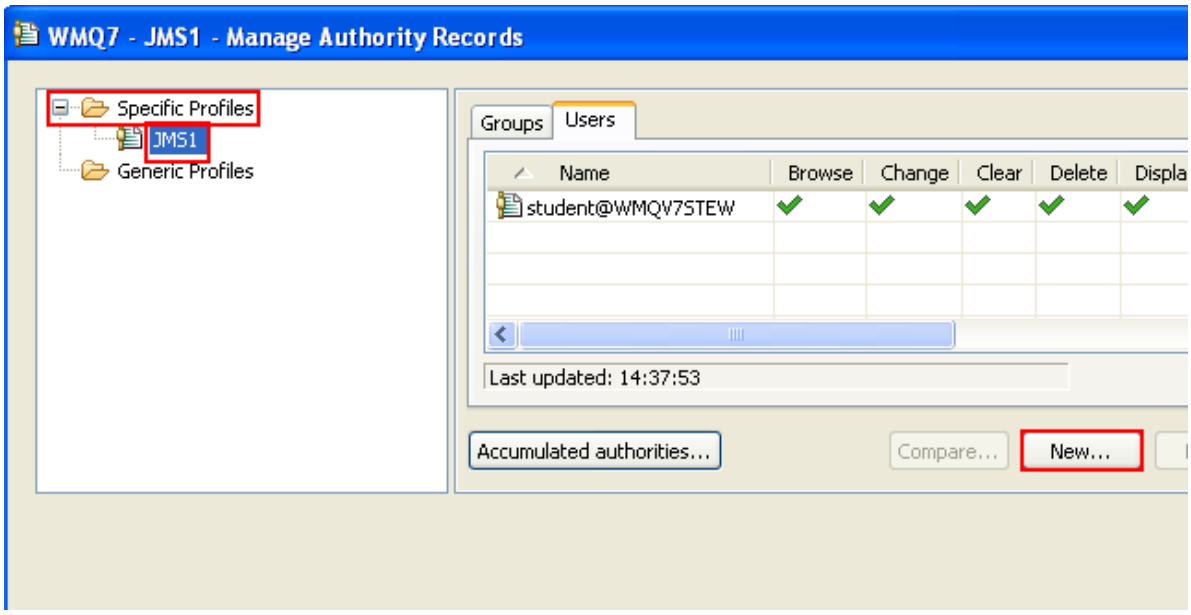
__6. Run the **amqsput** program again. **(amqsput JMS1).** User Brian is not authorized to write messages to that MQ queue.



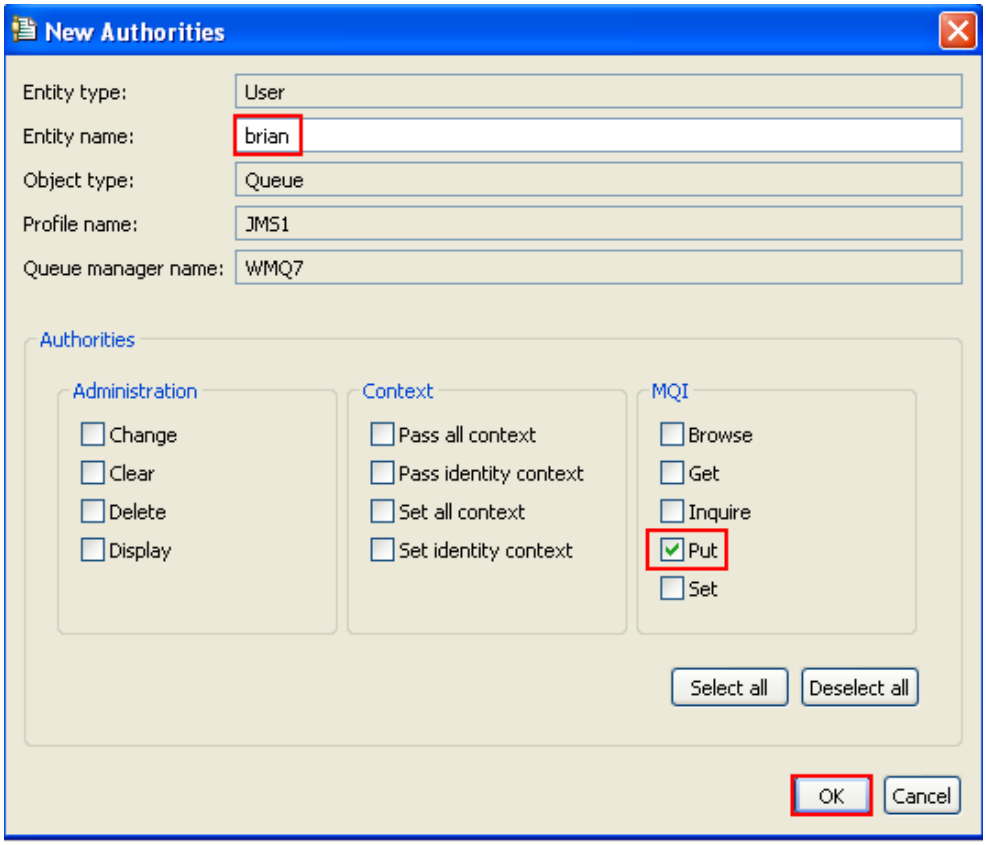
__7. There are several methods of administering Object Authority Manager permissions; programmatically, from the command line, or using the MQ Explorer. You will use the MQ Explorer. Open the MQ Explorer and then click on **Queues**. Right-click on the queue called **JMS1,** then on **Object Authorities→Manage Authority Records…**



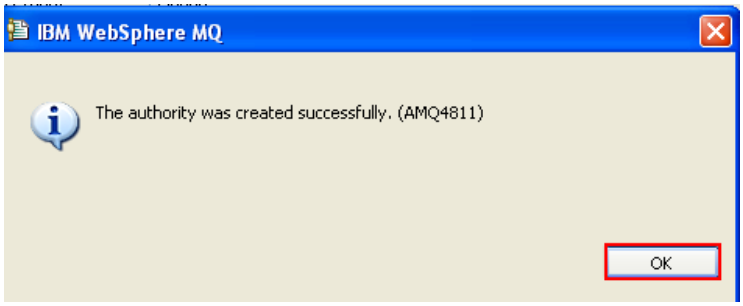
__8. Expand **Specific Profiles**, and then click on **JMS1.** Make sure that the **Users** tab is selected. Now click on the **New…** button



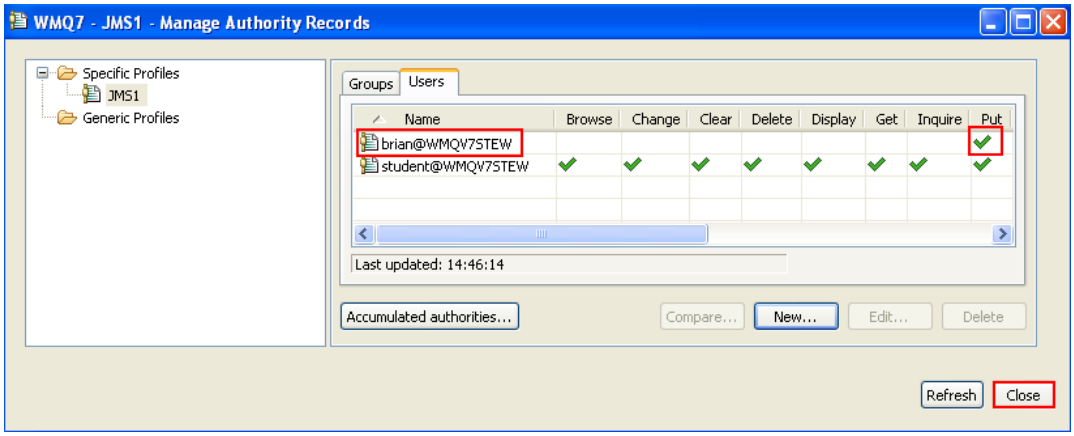
__9. You are going to set up a user entry for the user id **brian.** Type **brian** as the Entity Name. Under **Authorities** click next to **Put**. Then click on **OK.**



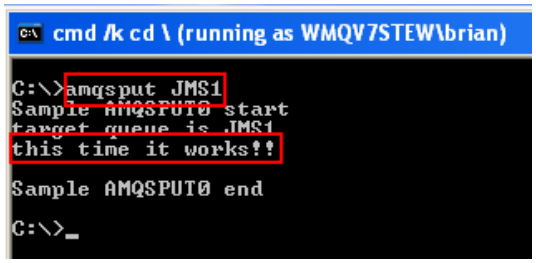
__10. Click on **OK** to close the confirmation window.



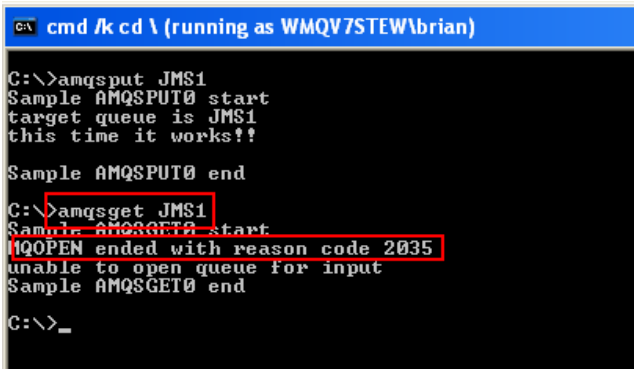
__11. Notice that a user record has been added with Put authority for user brian. Click on **Close** to close this window.



__12. Go back to the runas brian command window and type in **amqsput JMS1.** This time it should be successful. Enter some data; then press Enter again to stop the program.



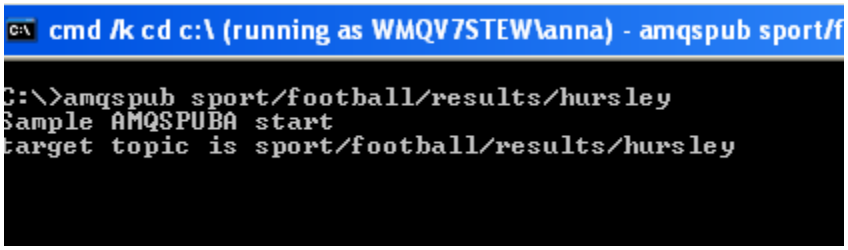
__13. You gave the user id brian the ability to put messages to queues, but not to read or browse messages. In the runas brian command window type in **amqsget JMS1**. This should fail with a return code 2035, **MQRC_NOT_AUTHORIZED.**



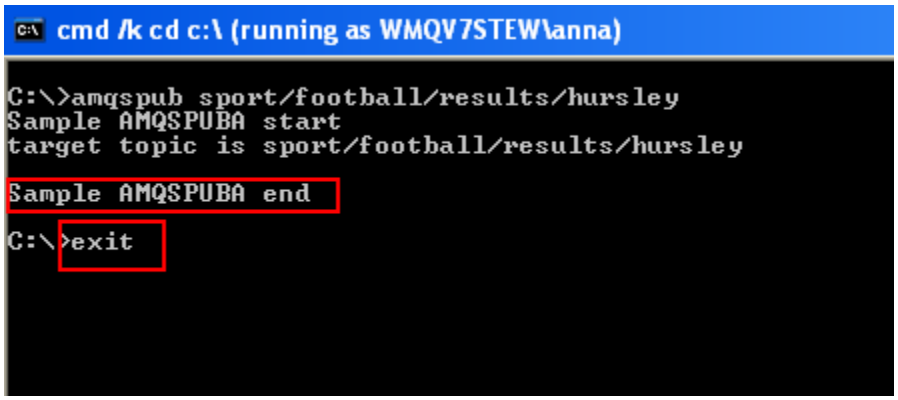
This concludes this portion of Lab 4.

4.5 Pub/Sub (Topic) Security

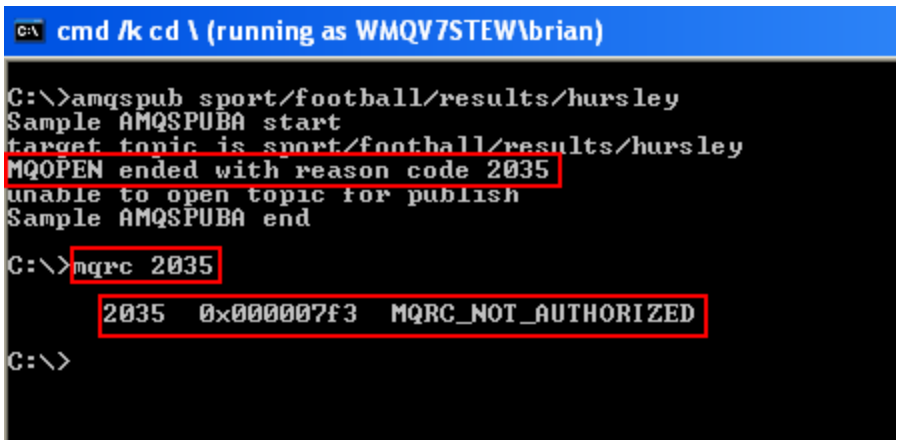
__1. You are going to do the same tests using Pub/Sub security to secure topics. As a test, try to publish to topic sport*/football/results/hursley*. In the "**runas**" window for Anna, type **amqspub sport/football/results/hursley** and press Enter. This should be successful, as Anna is a "superuser".



__2. Press Enter to end the amqspub program. Then type in **Exit** to close the runas Anna window.

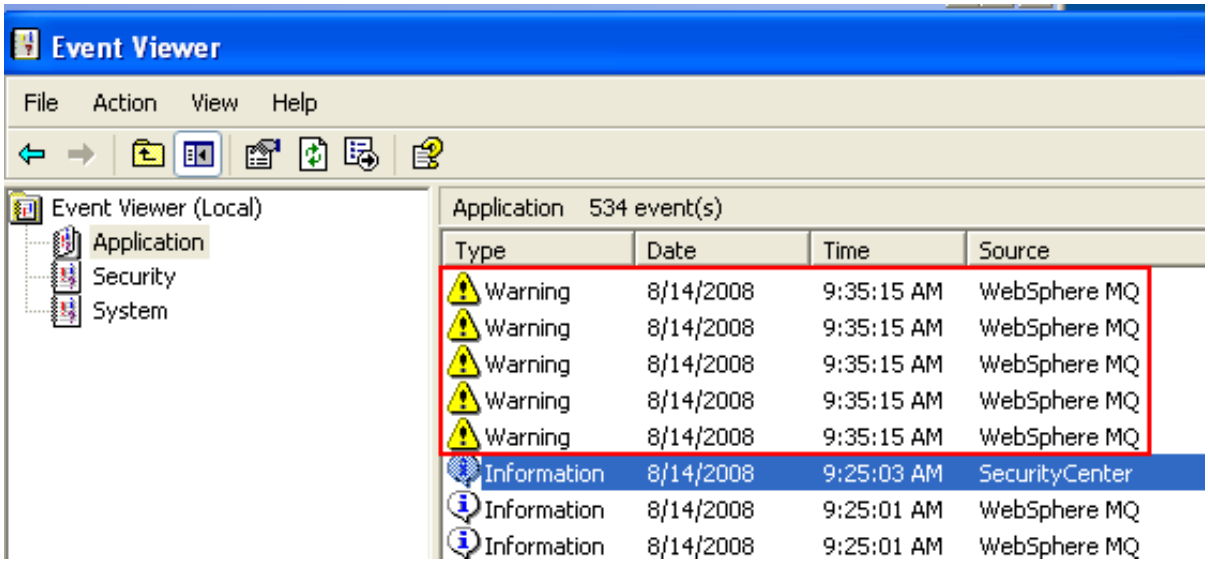


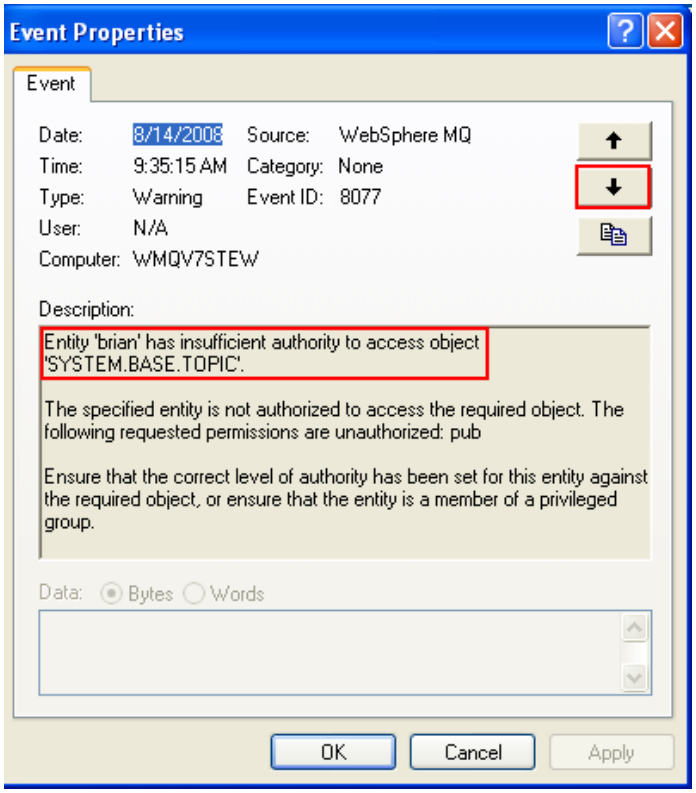
__3. Now try a userid that is not authorized to publish to a topic. Return to the **Runas_brian.cmd**. Again, try to publish to topic *sport/football/results/hursley*. In the window type **amqspub sport/football/results/hursley** and press Enter. This should end with a 2035 return code. Type in the command **mqrc 2035** and you should see that this is a security error.



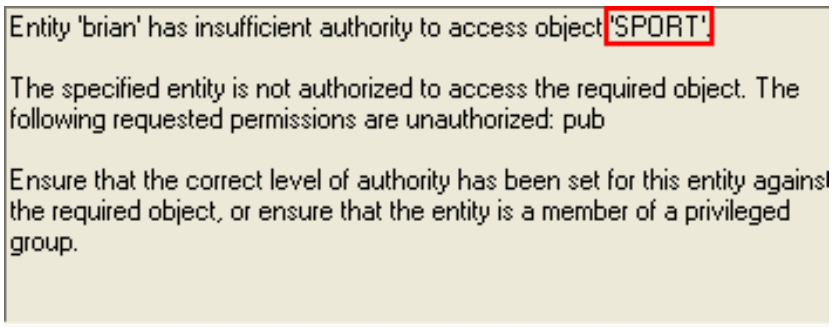
__4. You will see more interesting information in the Windows event viewer. To open the event viewer find the shortcut in the Start Menu.

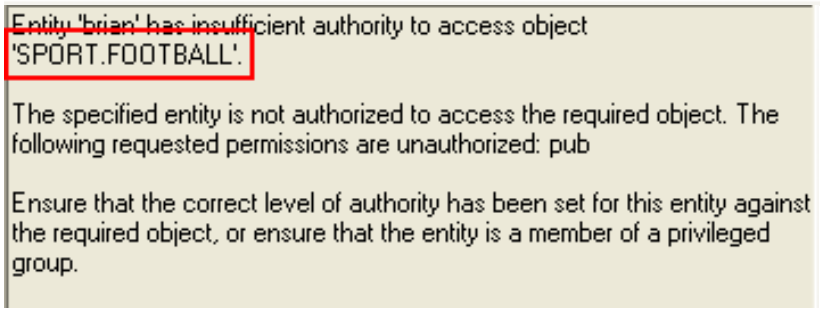
__5. You will see a number of warnings produced because of the 2035 return code. Double-click on the first warning to display its contents

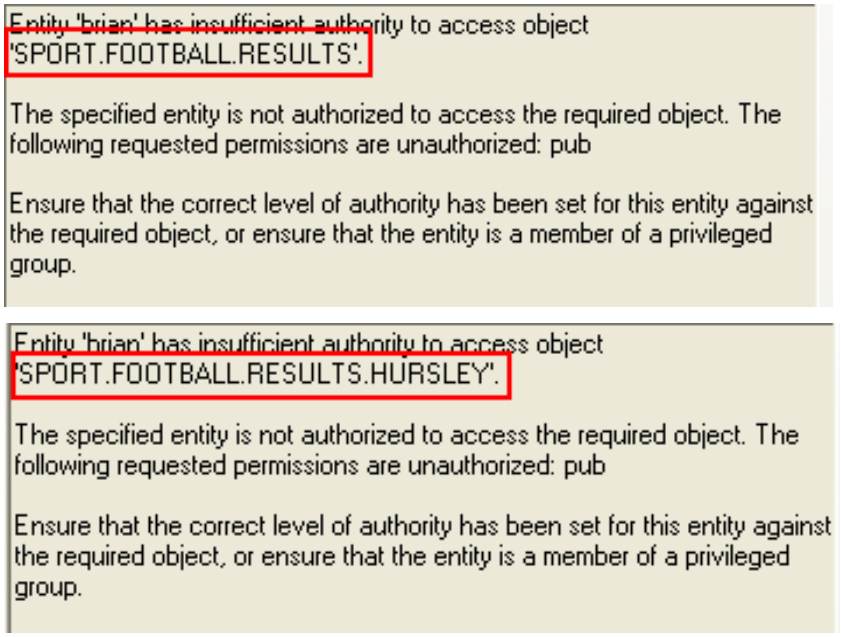




__6. **SYSTEM.BASE.TOPIC** is the root topic object for the topic tree as a whole. Click on the down arrow to browse through the remaining warnings.





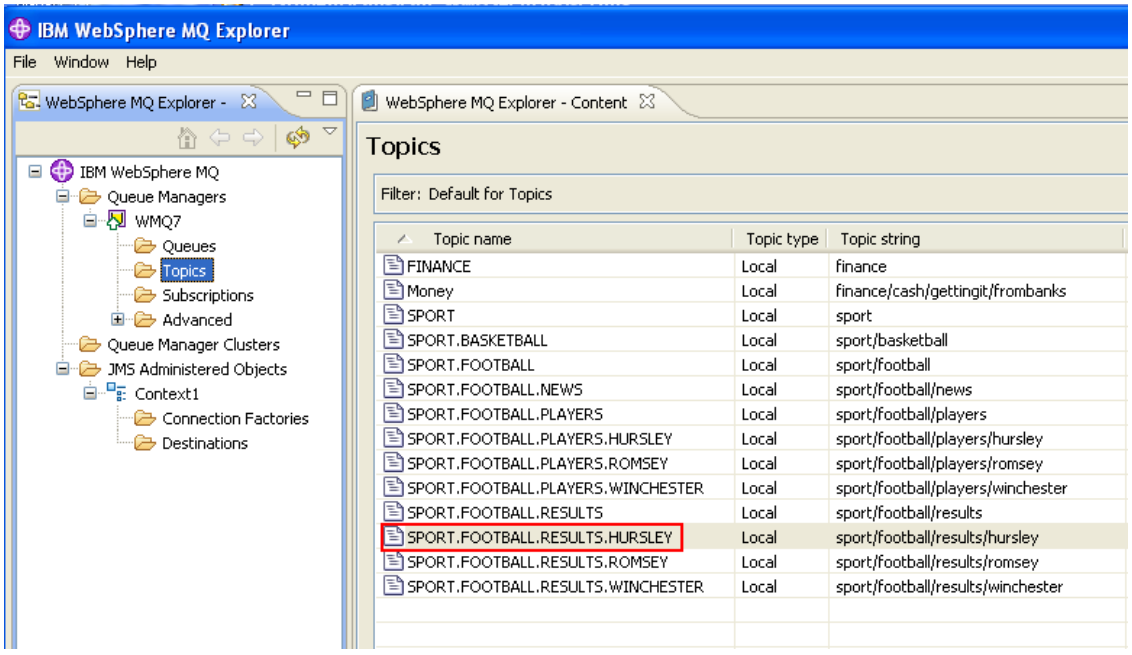


This concludes this portion of Lab 4.

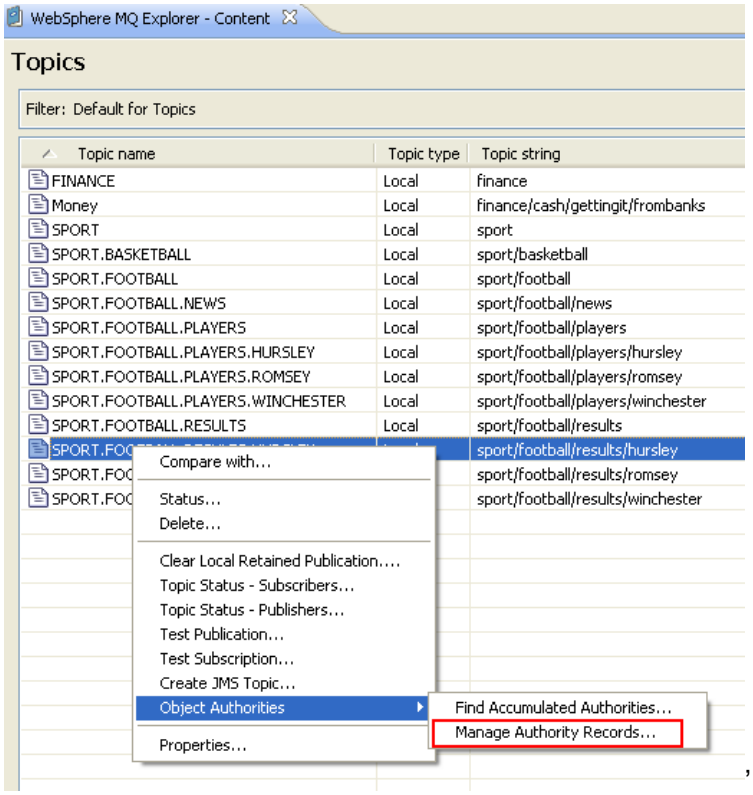
4.6 Using the MQ Explorer to manage security

User Brian needs access to this topic! So in the next section you will use the MQ Explorer to create the permissions that Brian needs to access the topic **sport/football/results/hursley**.

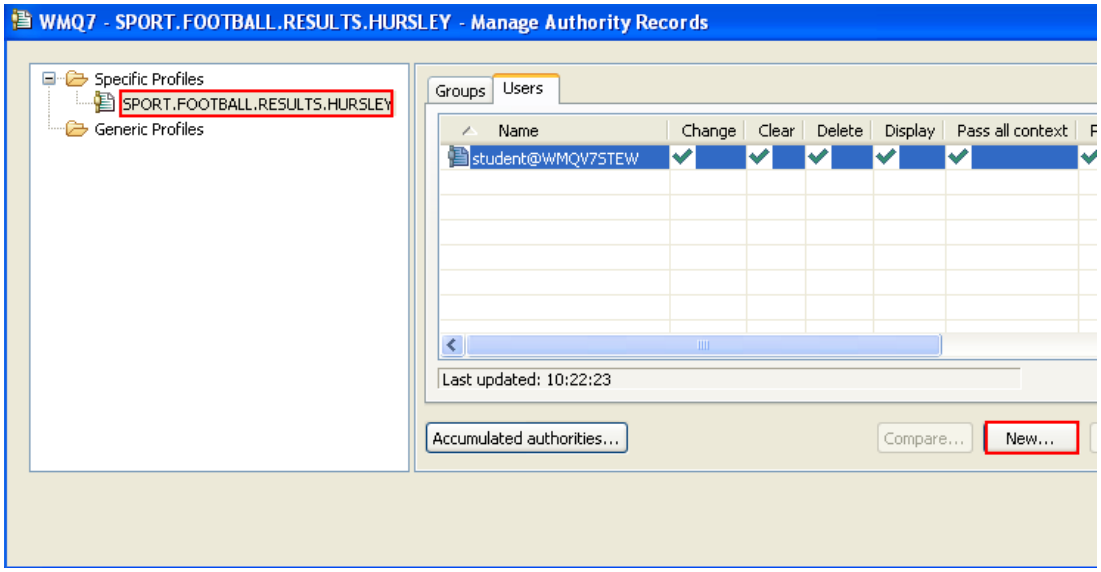
__1. Open the MQ Explorer and click on **Topics**



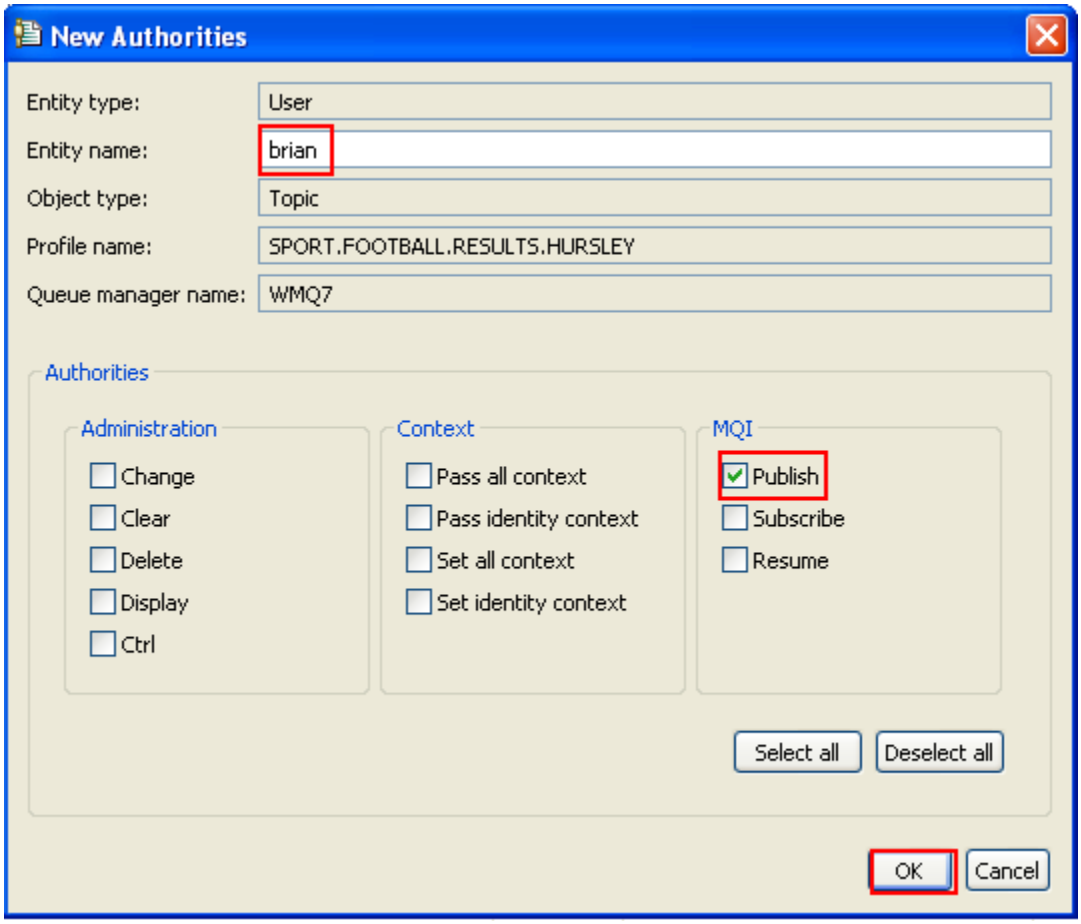
__2. Right-click on the topic name **SPORT.FOOTBALL.RESULTS.HURSLEY**. Select **Object Authorities→Manage Authority Records…**



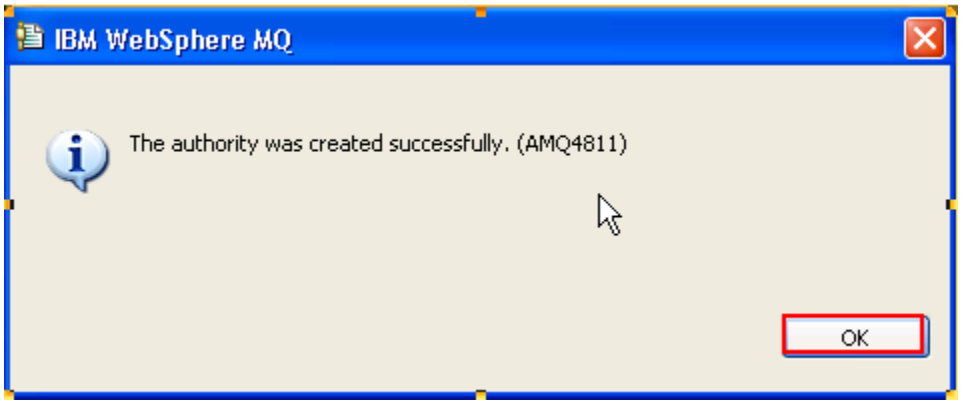
__3. A new window will open showing groups and users for the specific profile. Click on the Specific Profile **SPORT.FOOTBALL.RESULTS.HURSLEY**. Now click on the **Users tab**. Finally, click on the **New**… button. You are going to create a user record for the user id brian.



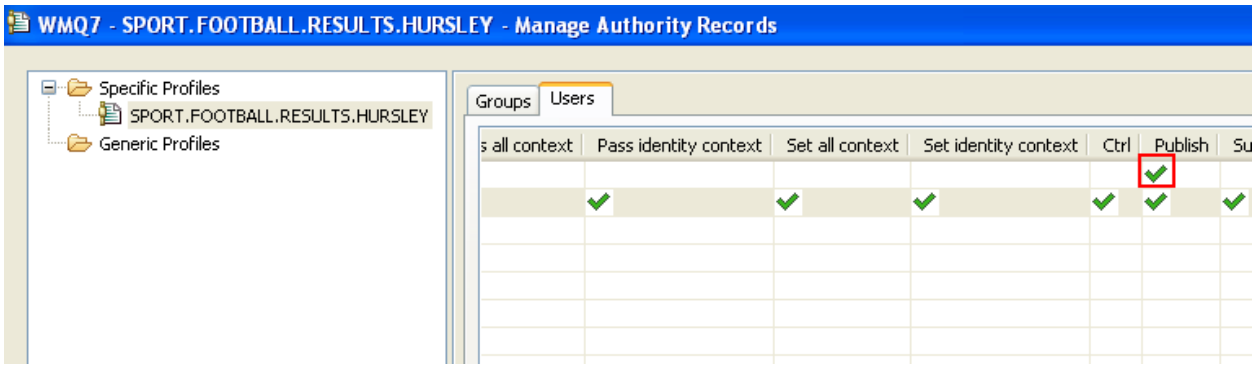
__4. Enter **brian** as the user name. Under **Authorities→MQI** click on the **Publish checkbox**. Then click on **OK**



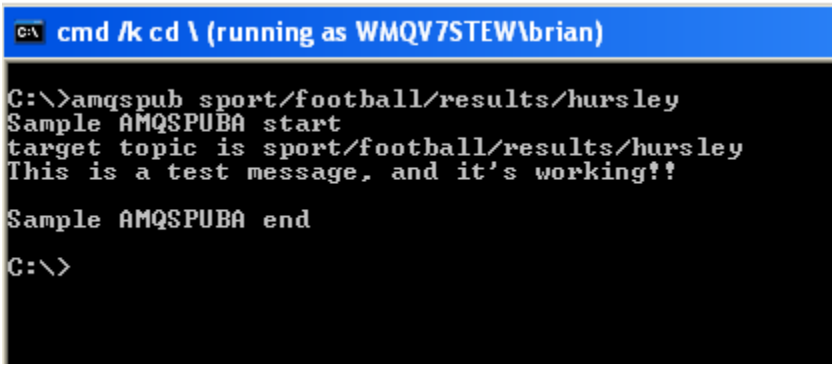
__5. Click OK to dismiss the confirmation box.



__6. You have now built a security tree where Brian is granted authority to publish football results. Scroll to the right to see the green checkmark under the Publish column.



__7. Test that Brian can publish to the **sport/football/results/hursley** topic (double-click on the **Runas_brian.cmd** script, enter the **passw0rd** then enter the command **amqspub sport/football/results/hursley** and type a test message to the Topic



This concludes the Publish/Subscribe lab.

Lab 5 WebSphere MQ HTTP Bridge

In this lab you will be introduced to the WebSphere MQ – HTTP Bridge. You will exercise some of the features of the Bridge and see how easy it is to enable browser-based applications to access MQ resources.

This lab does use some external tools that are not shipped as part of WebSphere MQ. For example, you will employ a standalone HTTP Listener program available from IBM® as a SupportPac™ (SupportPac MA94) – this is a WebSphere MQ product extension, but is not shipped with the product. You will also use cURL. cURL is a freeware tool that provides a command line facility for transferring files using URL syntax. The strong point of cURL is the number of data transfer protocols it supports. You will use it to submit HTTP requests as part of this lab.

In this lab you will:

- Start the stand-alone HTTP Listener program.
- Use cURL to send HTTP requests to the listener.
- See the options and responses available to the listener.
- Use a sample JavaScript program to put and get messages, illustrating a zero footprint MQ client.

The http-listener program and all other resources needed in the lab are supplied in the c:\Student\Lab_HTTP directory.

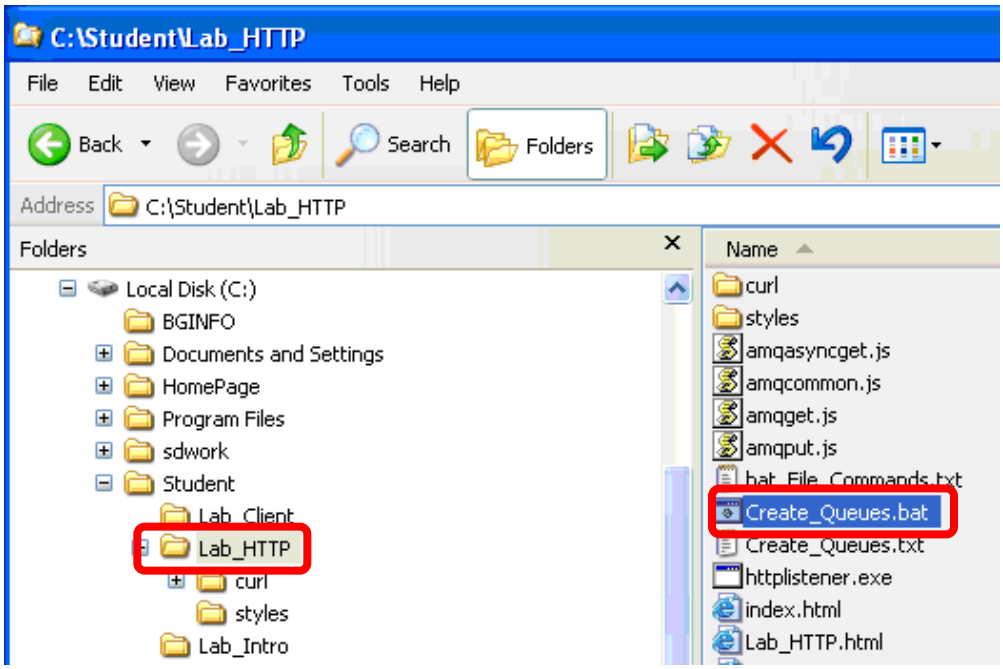
This lab assumes that queue manager WMQ7 exists. You were directed to create this during Lab 1.

5.1 Create the Required Queues

In Lab 1 you used the MQ Explorer to create some queues. In this lab you will use a command script file for that task so you can see another option that can be used to do this administrative work. This approach uses an MQ utility called **runmqsc**. This program will take a series of MQ administrative commands and execute them. Those commands can come from the keyboard or from a file. In this case a file will be used.

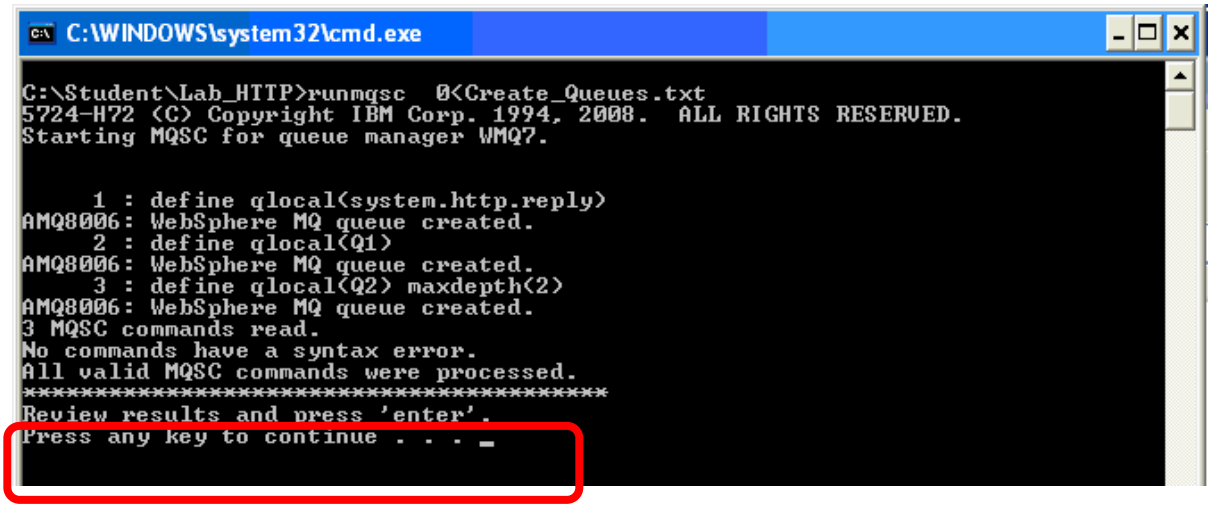
__1. Launch **Windows Explorer** using the icon in the system tray (if you have Windows Explorer already launched then bring it into active view).

__2. Navigate to **C:\Student\Lab_HTTP** and double-click on the **Create_Queues.bat** file.



__3. The following command is issued: **runmqsc < Create_Queues.txt** The "<" indicates that the commands will come from a file and that the path to that file follows. You don't have to enter a queue manager name for this command because you specified that WMQ7 was the default queue manager on this system when you created it. If that were not the case the name of the queue manager would follow runmqsc immediately (i.e. runmqsc WMQ7 < Create_Queues.txt).

Below you see the results from executing the command….hit the Enter key after reviewing.

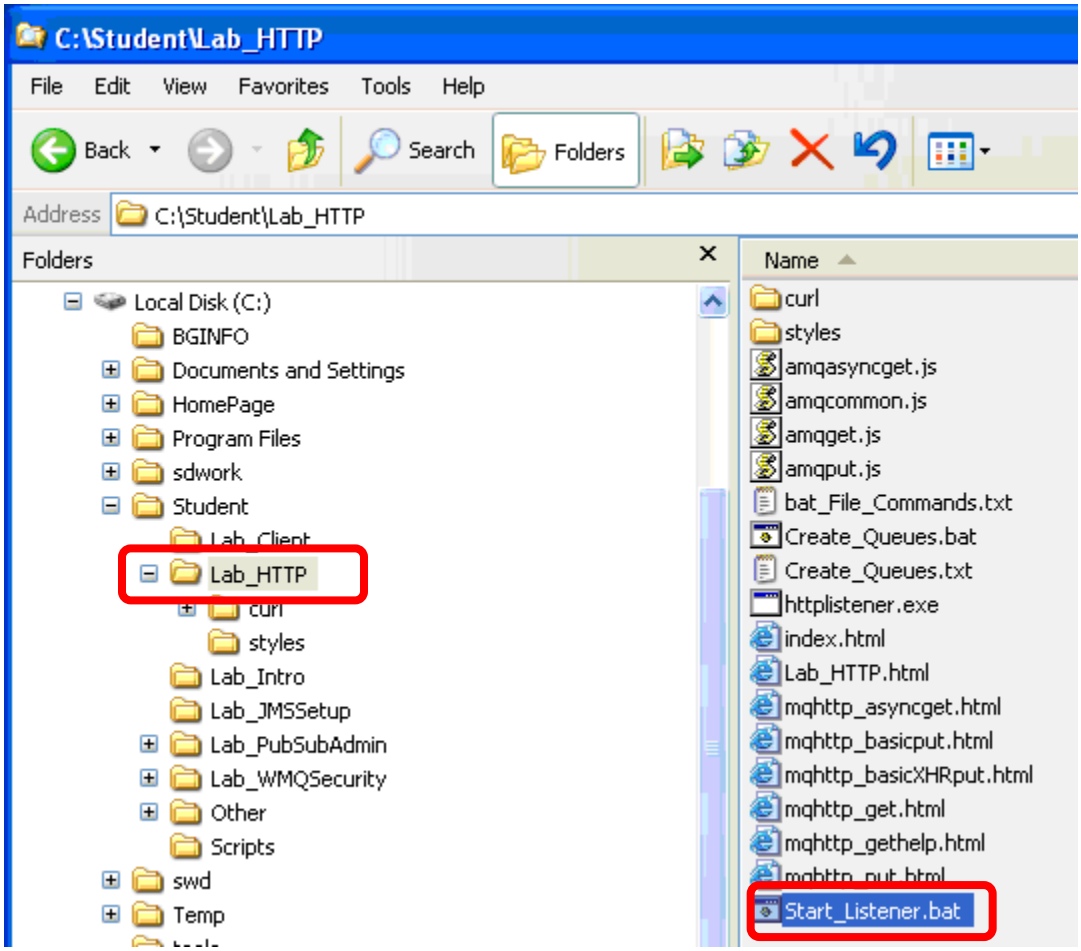


Note that the **Q2** queue has been assigned a **maximum queue depth of 2**.

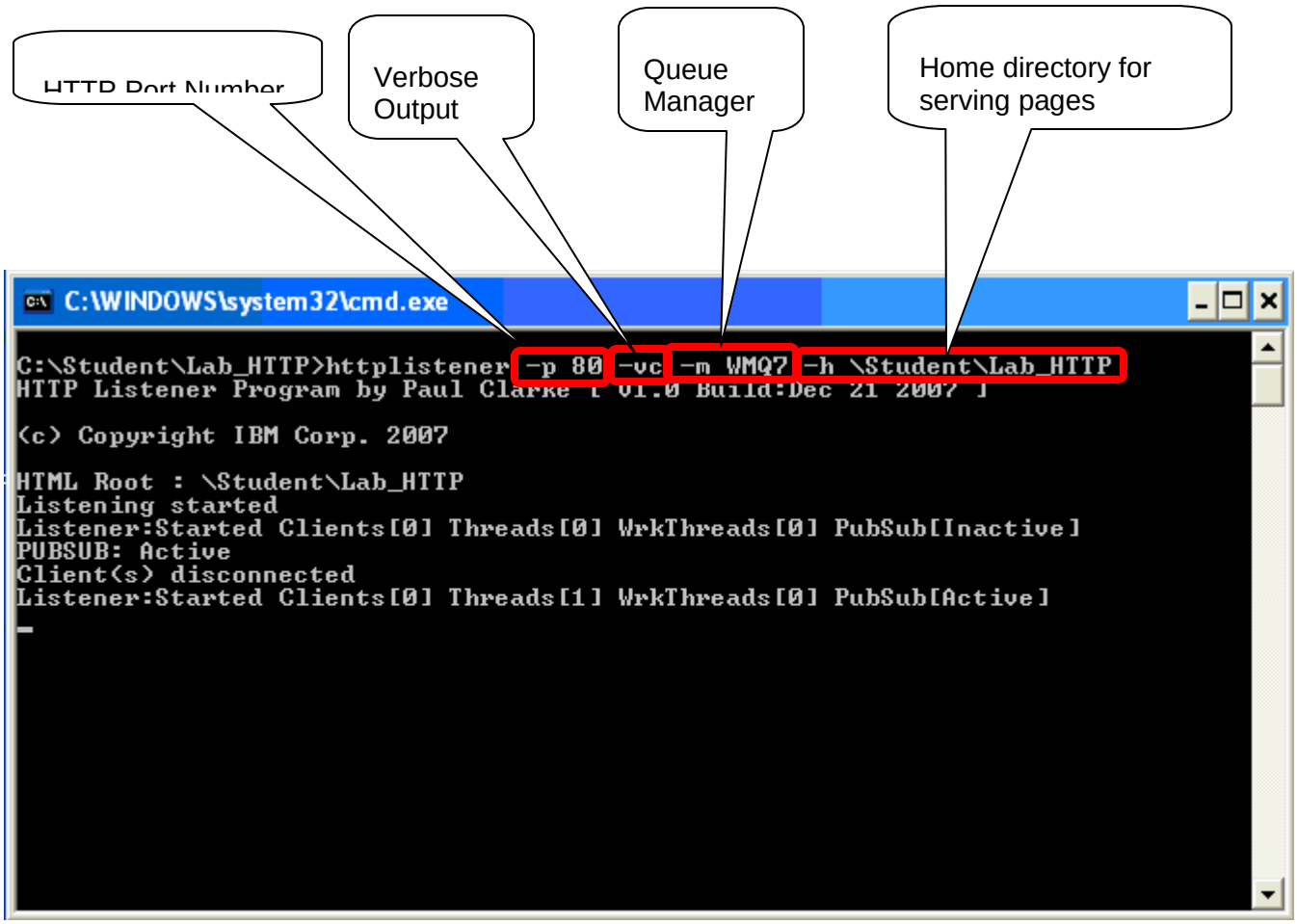
This concludes this portion of Lab 5.

5.2 Start The HTTP Listener

__1. Double click on the **Start_Listener.bat** file.



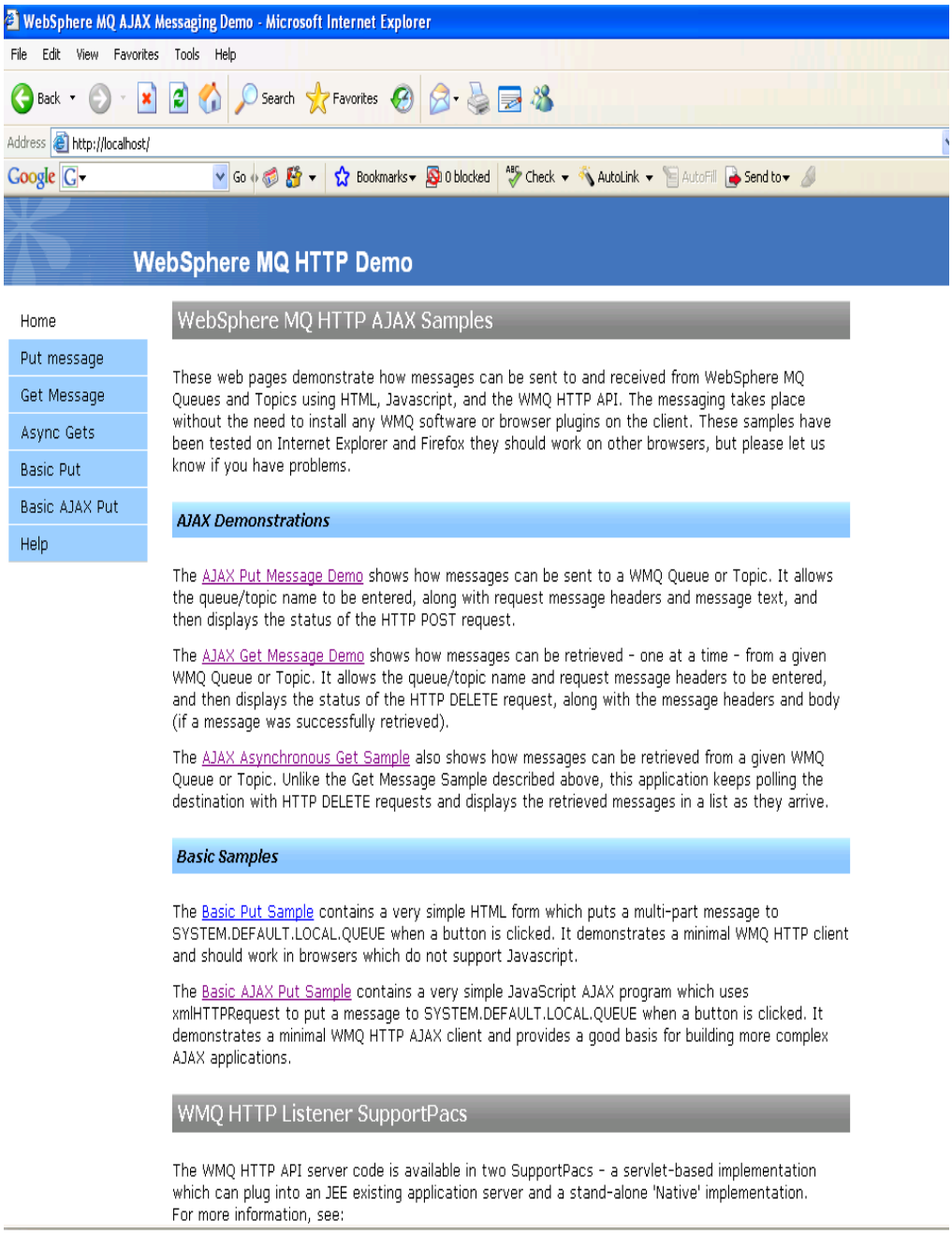
__2. The following command is executed…..review its output. Note that this command window will stay open since the MQ HTTP Listener is going to continue to run in this session.



__3. This will start a listener on HTTP port 80, connecting to queue manager WQM7. The root for serving web pages will be \Student\Lab_HTTP…

Do not close this command window!

__4. Verify that the listener is running by **opening a web browser** and pointing it to **http://localhost**. You should see the web page shown below.



__5. This indicates the listener is running. Close the browser.

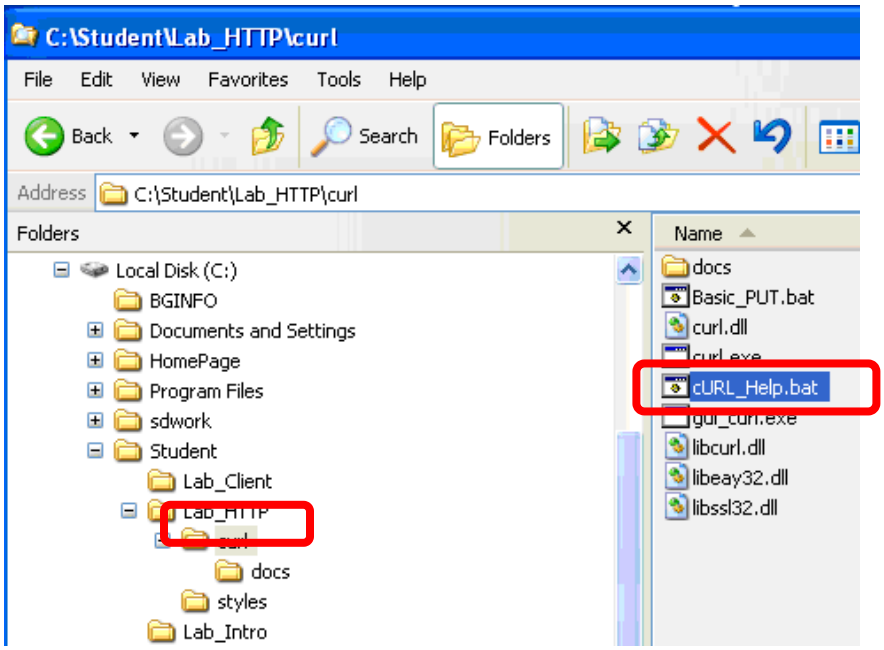
This concludes this portion of Lab 5.

5.3 Test the Listener Using cURL

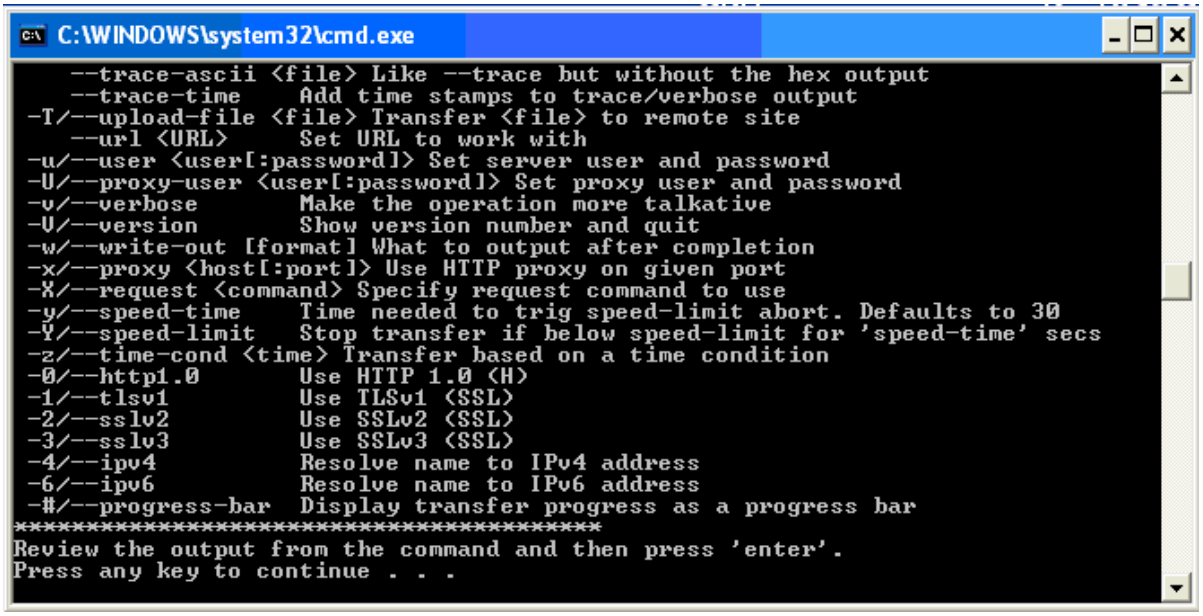
In this section you use the cURL program to send HTTP requests to the listener and examine the responses.

5.4 Posting Messages Using HTTP Post

__1. Navigate to the **cURL** folder and double click on **cURL_Help.bat**….



__2. The following command is run. **Examine the output** from the command and then hit the Enter key……

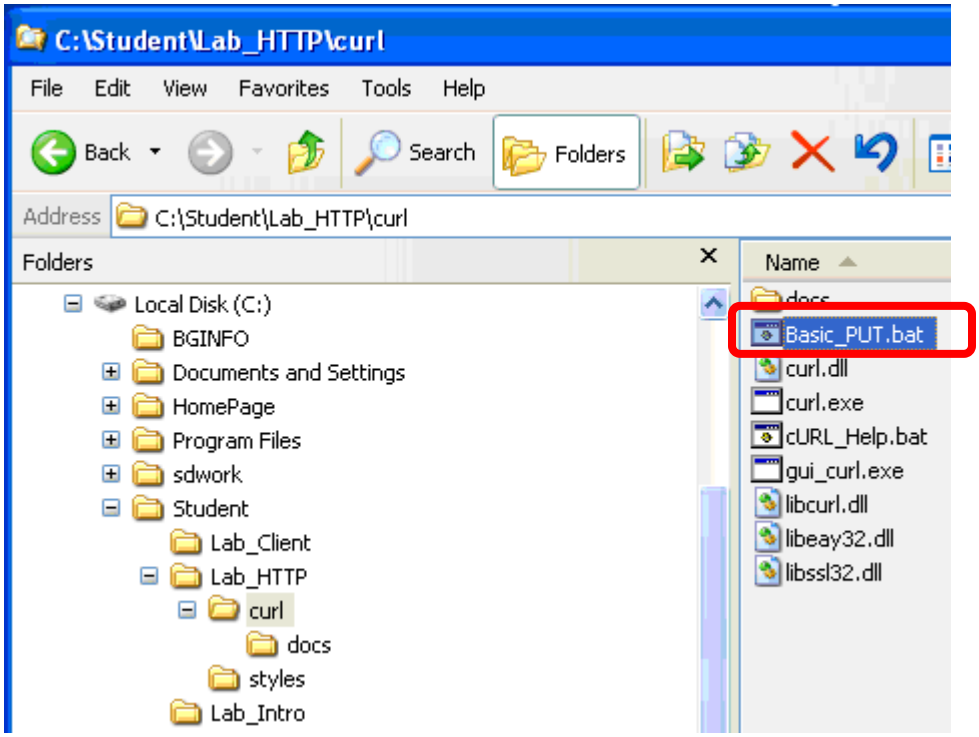


__3. You will use only a few of the options and these are shown below:

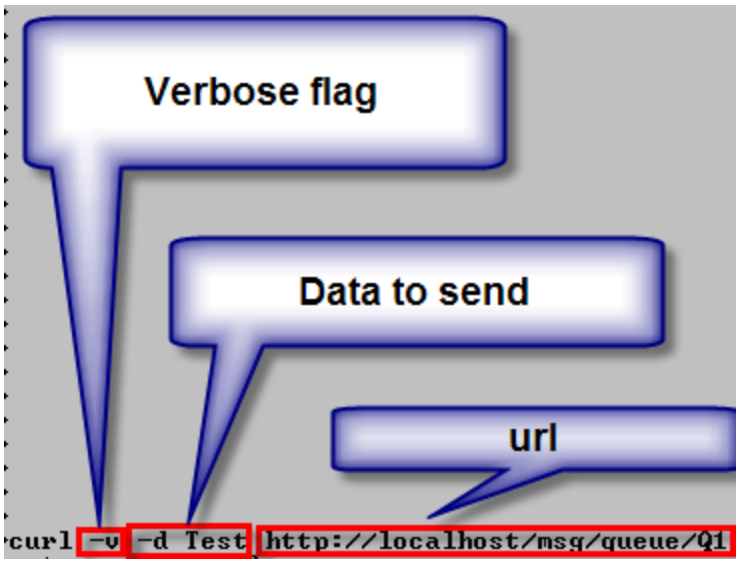
```
curl --help

Usage: curl [options...] <url>


-d/--data <data>   HTTP POST data (H)


-H/--header <line> Custom header to pass to server (H)


-v/--verbose       Make the operation more talkative
```

5.5 A Very Basic PUT

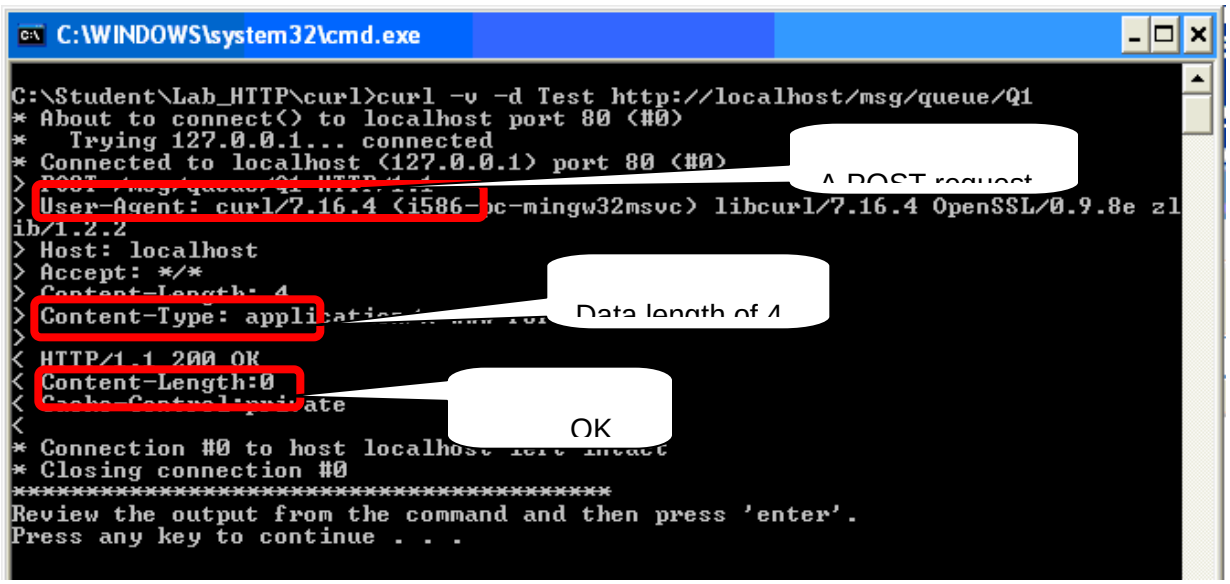
__1. Double click on the Basic_PUT.bat file…..



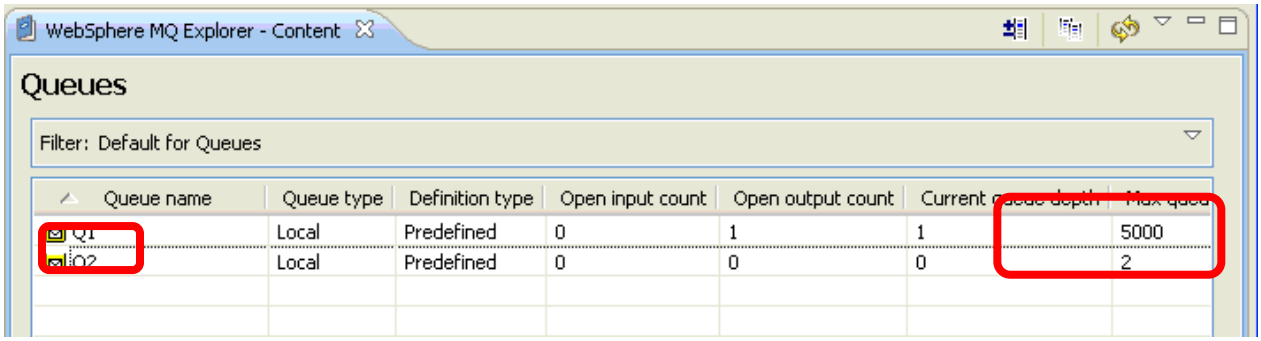
__2. The following command will be run.



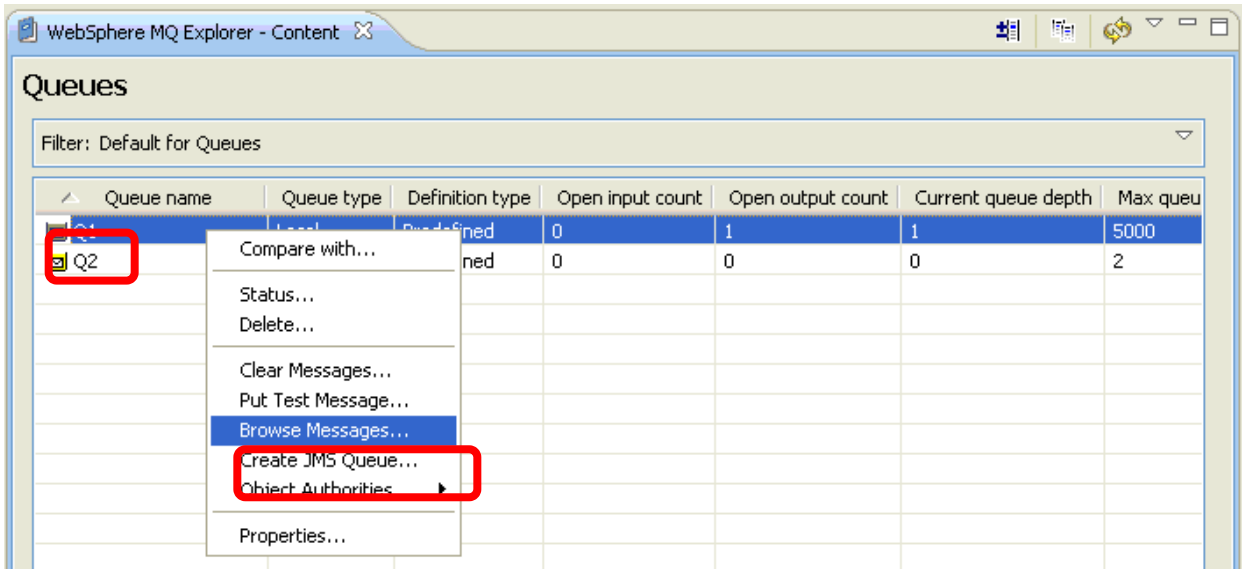
__3. Examine its output then hit the Enter key.



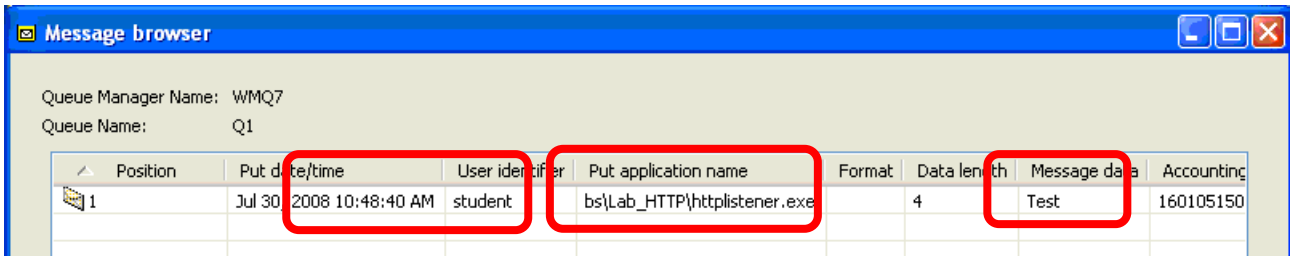
__4. Bring the **MQ Explorer** into active view. Note that you now have a message on **Q1**.



__5. Right click on **Q1** and select Browse **Messages**…



__6. View the summary data about the message and pay particular attention to the **Put date/time** and the **Put application name**.

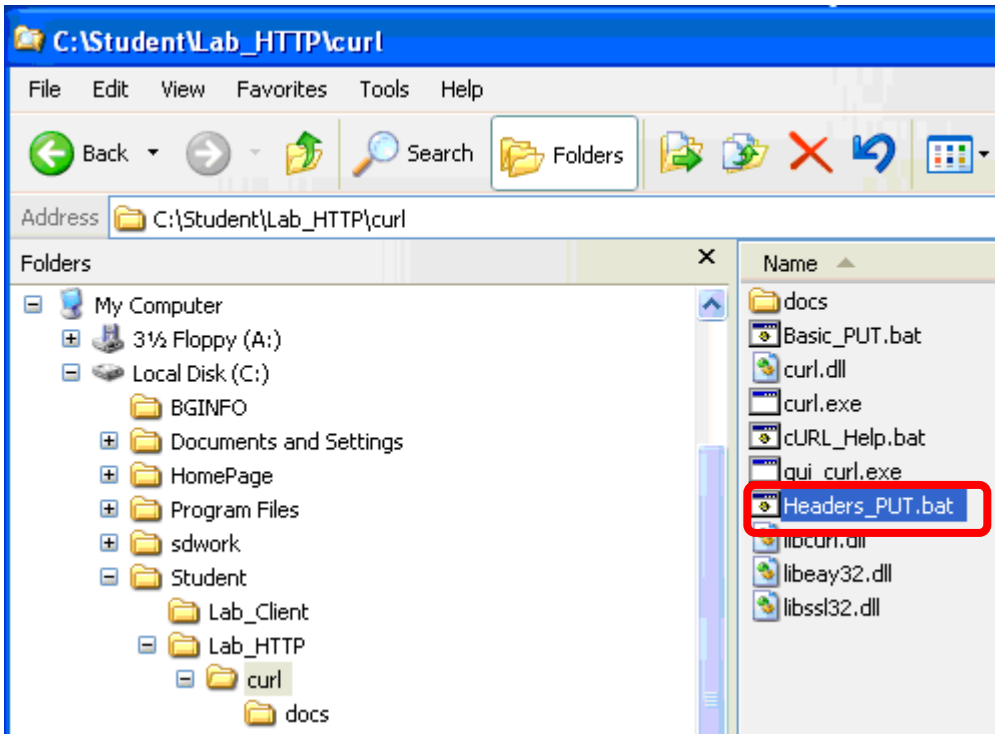


__6. Close the Message browser window.

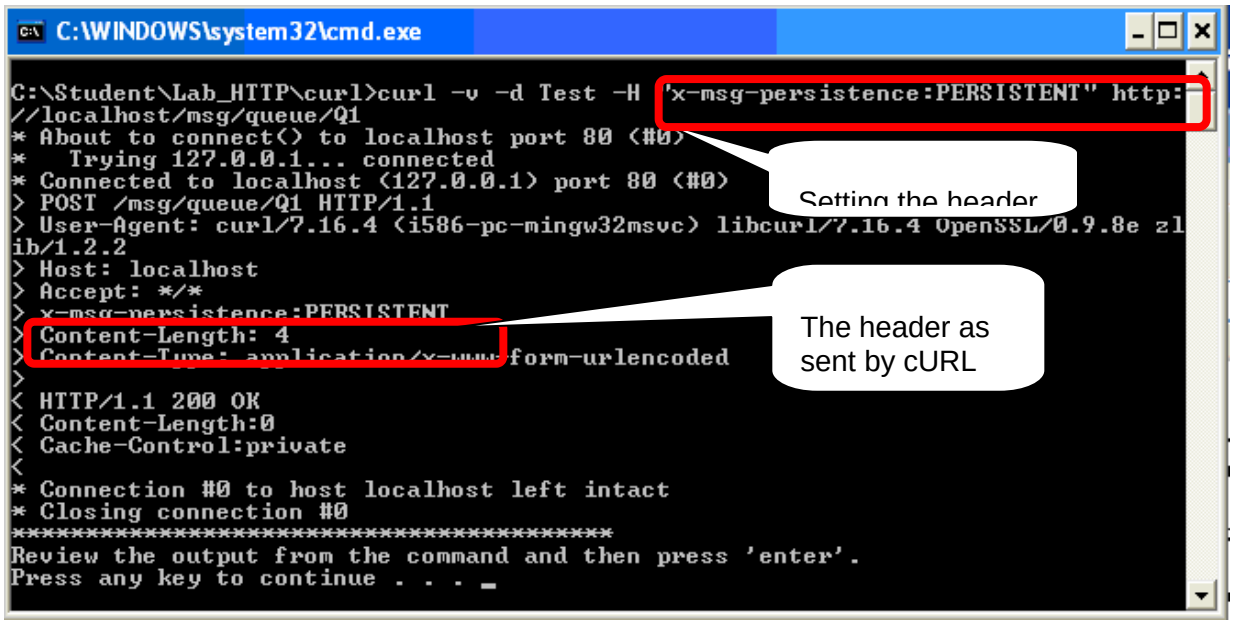
5.6 A PUT with an MQ Property Specified in the Headers

When using the WebSphere MQ – HTTP Bridge, MQ message header fields (MQMD) are conveyed in HTTP headers. This approach conforms to the HTTP specification. All HTTP headers used by MQ are prefixed with “x-msg-”. In this portion of the lab you will specify an MQ-specific header property for the HTTP message.

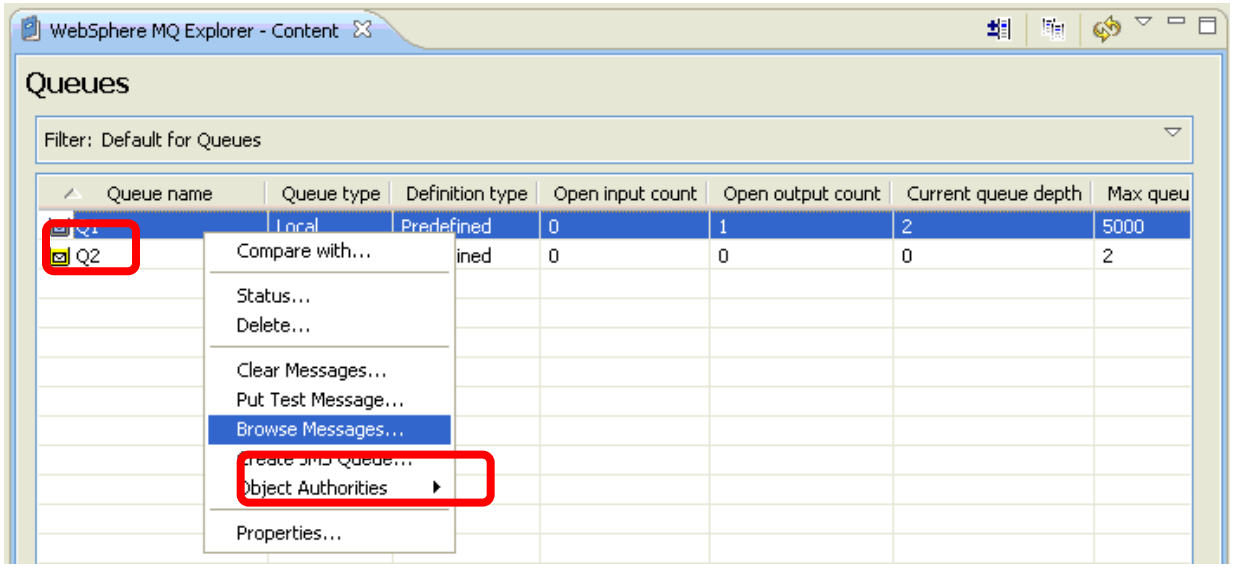
__1. Double click on the **Headers_PUT.bat** file.



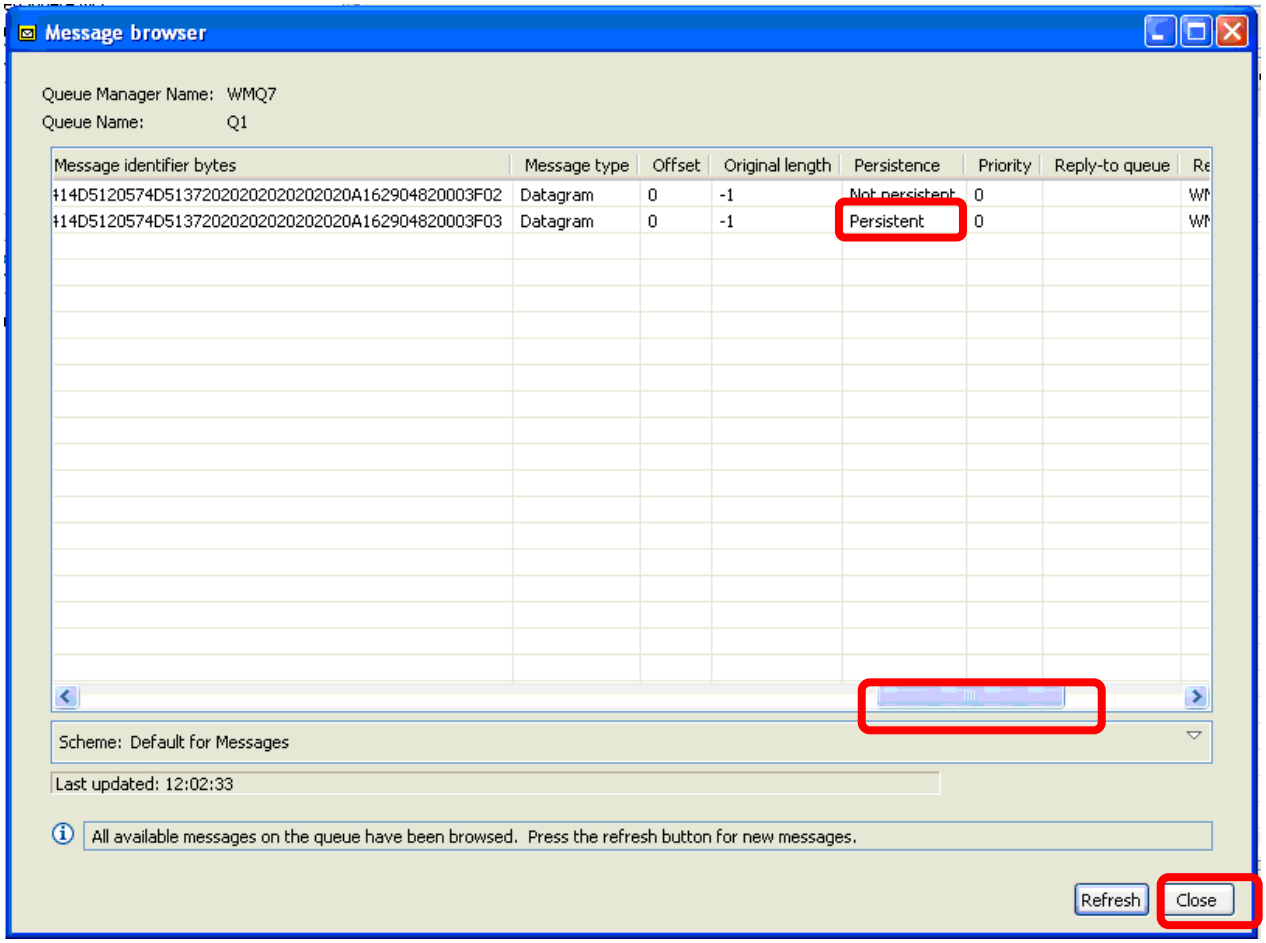
__2. The following command will be run. **Examine its output** and then hit the **Enter** key.



__7. Bring the **MQ Explorer** into active view. Note that the Current queue depth is now 2. **Right click** on **Q1** and select **Browse Messages…**



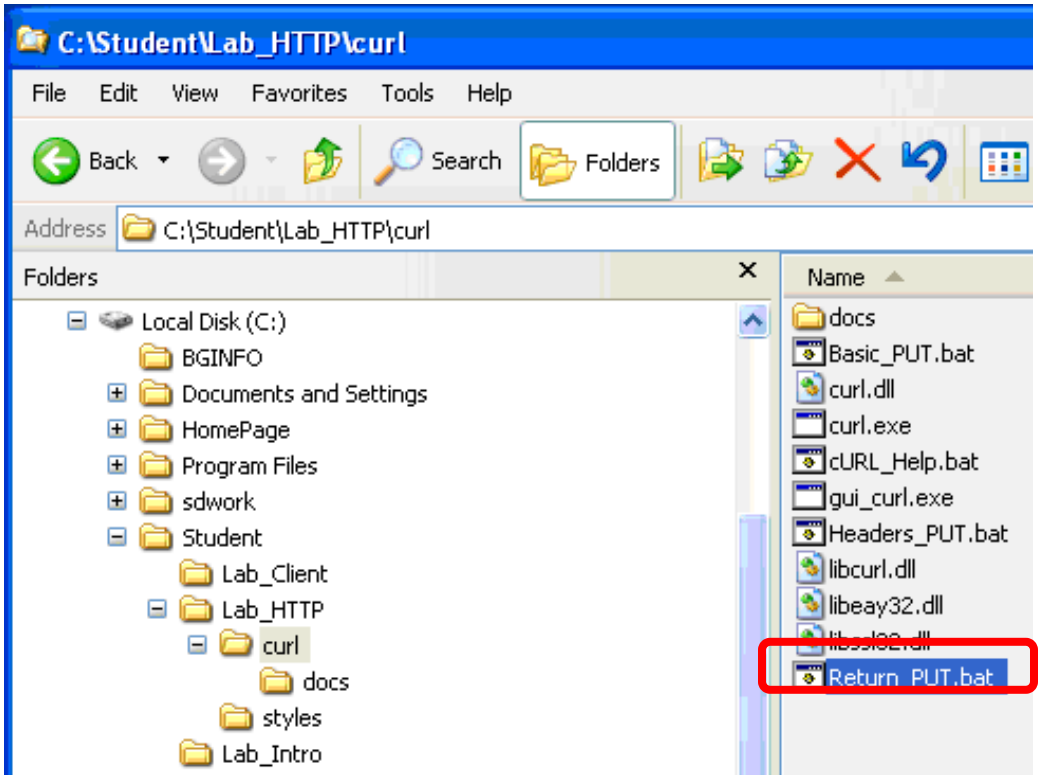
__8. Examine the second message to **verify that it is persistent**. You will need to **scroll** quite a bit to the right to see this message property in the display. Click the **Close** button.



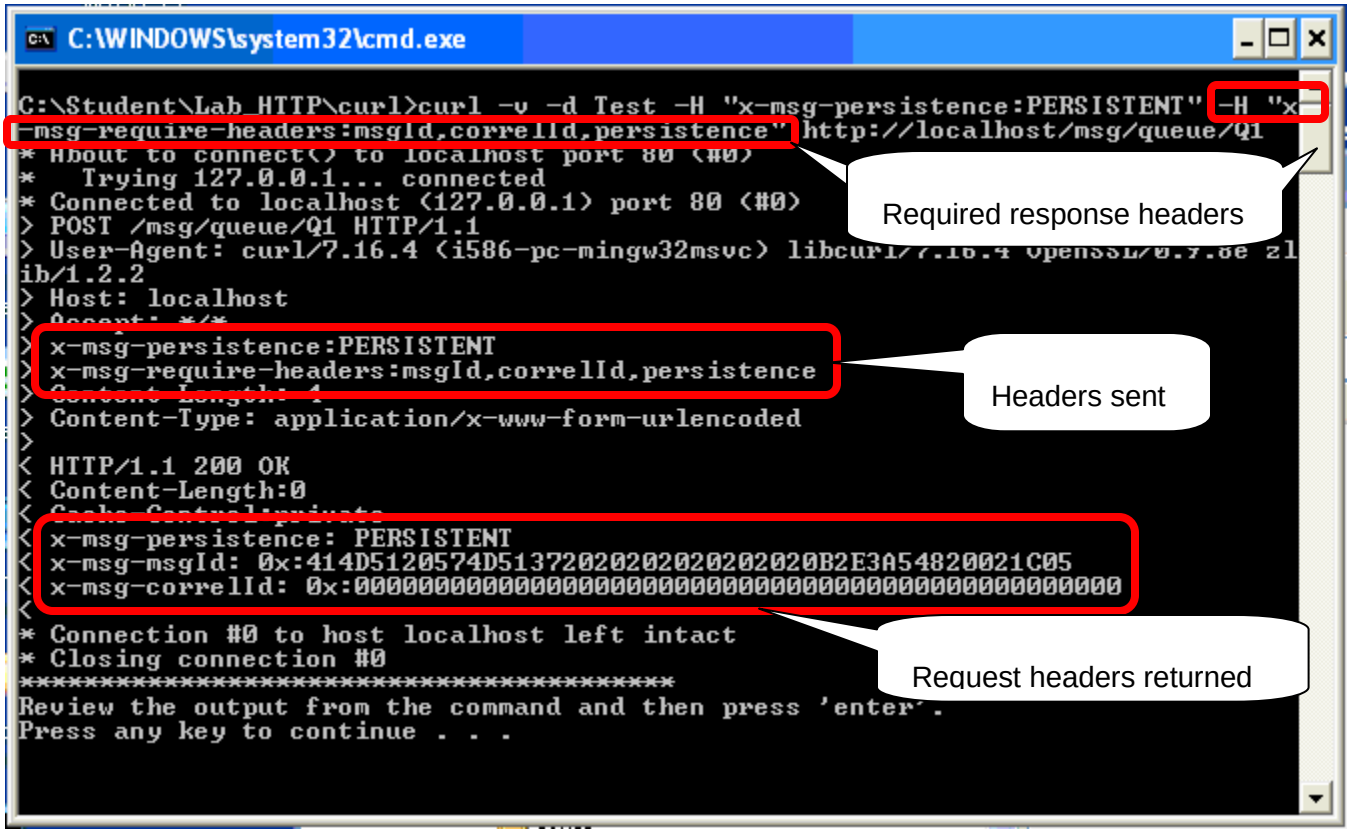
5.7 Requesting Information To Be Returned

cURL allows us to use the –H flag repeatedly to add as many headers to a message as you would like. In the next example you will use that capability to request certain information be returned to us in the response headers.

__1. Double click on the Return_PUT.bat file…..



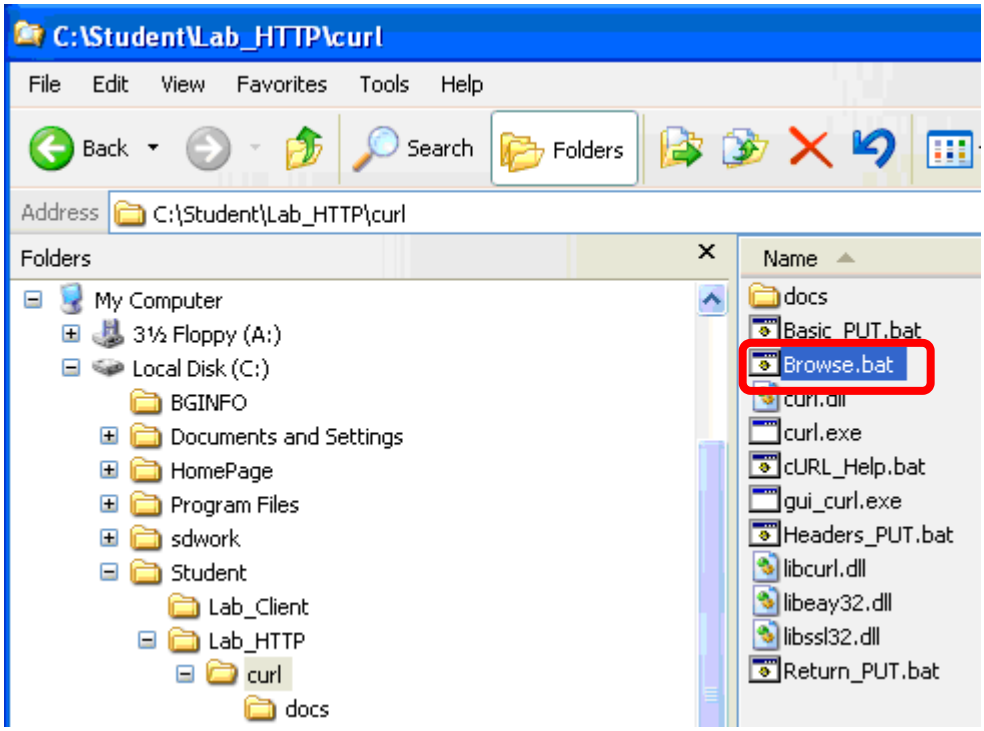
__9. The following command is run. **Examine its output** then hit the **Enter** key.



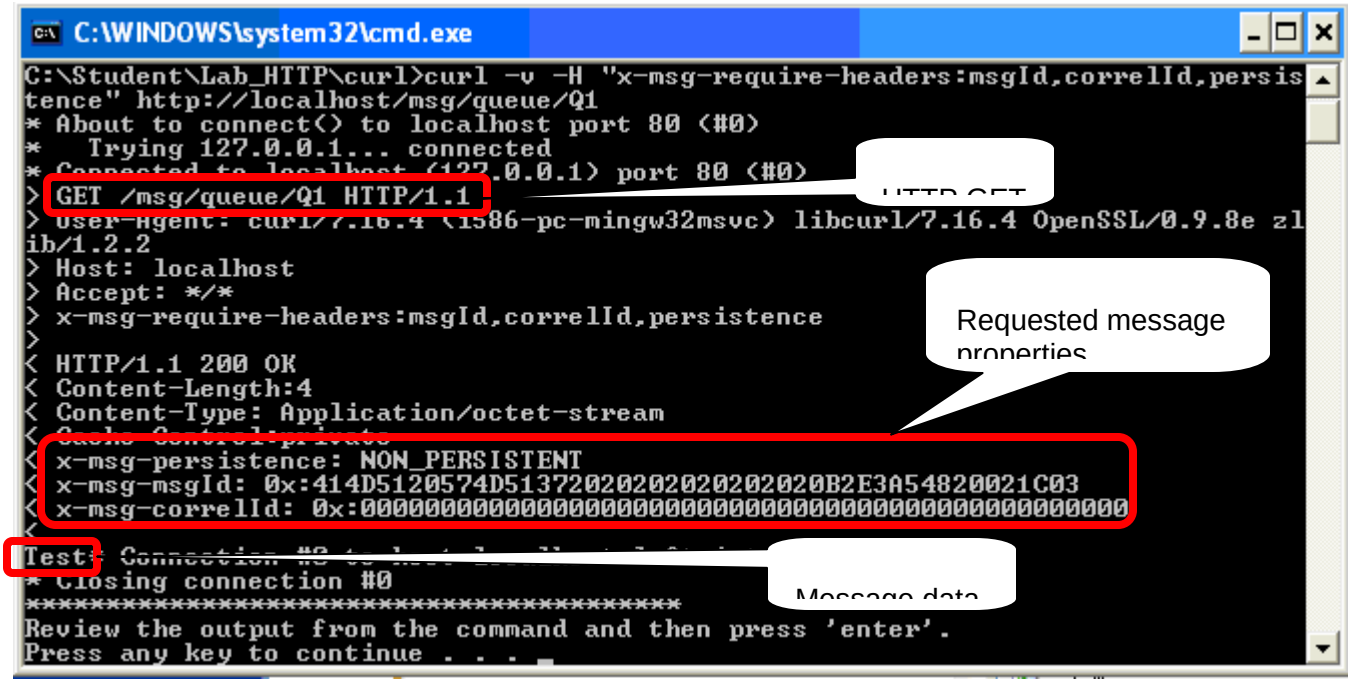
5.8 Using cURL To Browse Messages

If the –d flag is omitted (remember this flag indicates HTTP post data) cURL can be used to issue the HTTP GET command. The MQ – HTTP Bridge interprets an HTTP GET as an MQ Browse request…the message content is returned but the message is not removed from the queue….the Browse operation is not destructive.

__1. Double click on the **Browse.Bat** file…..



__10. The following command is run. **Examine its output** and hit the **Enter** key.



__11. Repeat the request a few times and note that the same message is returned with each invocation.

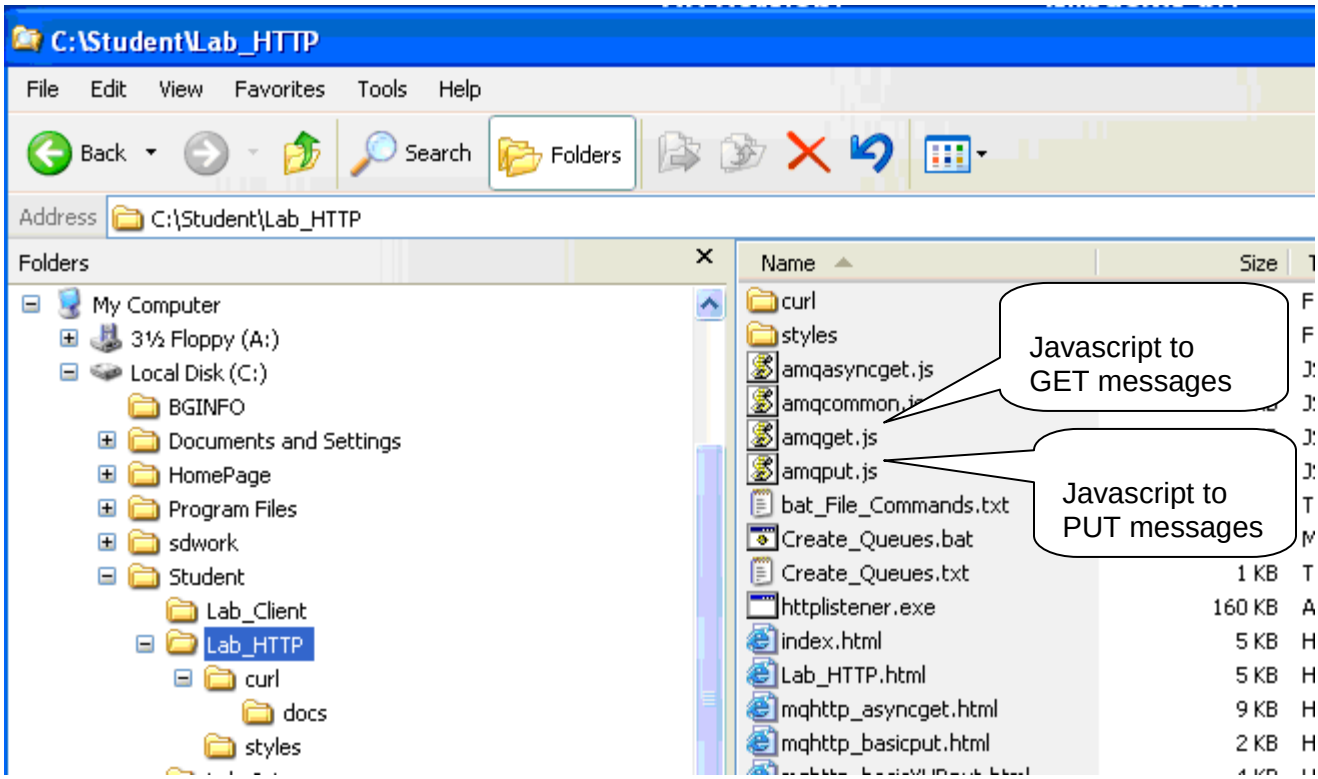
This concludes this portion of Lab 5.

5.9 Using the HTTP Bridge With JavaScript

Using cURL was instructive because it allows us to see how MQ header properties can flow over the HTTP communication stream without the need for any MQ-specific code. But cURL is not likely to be used by end users.

This part of the lab shows how a simple JavaScript can be used to put and get messages to and from queues.

The C:\Student\Lab_HTTP directory is the home directory for the HTTP listener to serve pages from. In that directory are two simple JavaScript pages, amq*put.js* and amq*get.js*.



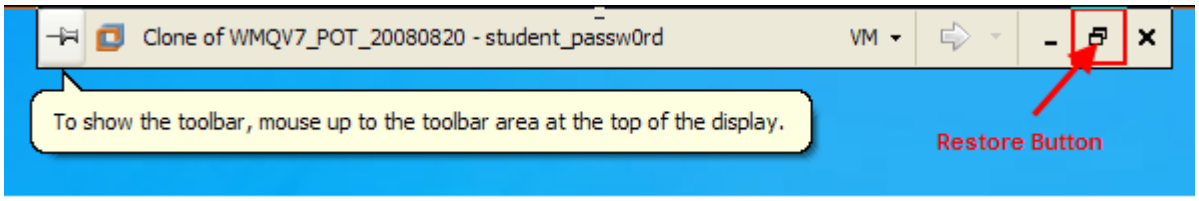
If you understand JavaScript you may want to look at these files to understand them.

One critical point is that a client needs no MQ libraries or code installed to run these JavaScripts.

You demonstrate this by running the next part of the lab **NOT ON THE VMWARE IMAGE** but on the **NATIVE OPERATING SYSTEM OF YOUR COMPUTER** which does not have any MQ code installed on it.

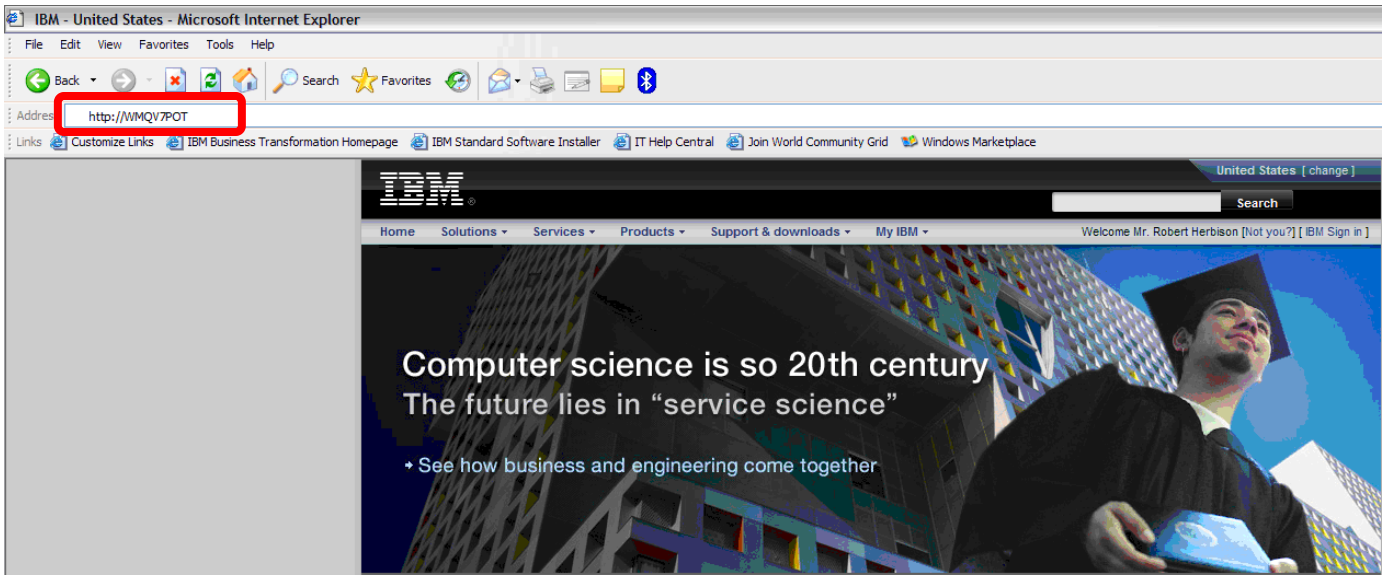
So you will now **switch to the desktop** of the **native system** on your computer.

__1. At the top of the window, find the VMware Toolbar. It may be hidden, or “unpinned”. Place your cursor near the top of the display, and the toolbar will automatically scroll down. Then click on the **Restore button.** Your display should change. You can now more easily switch between the native operating system and the VMware image.

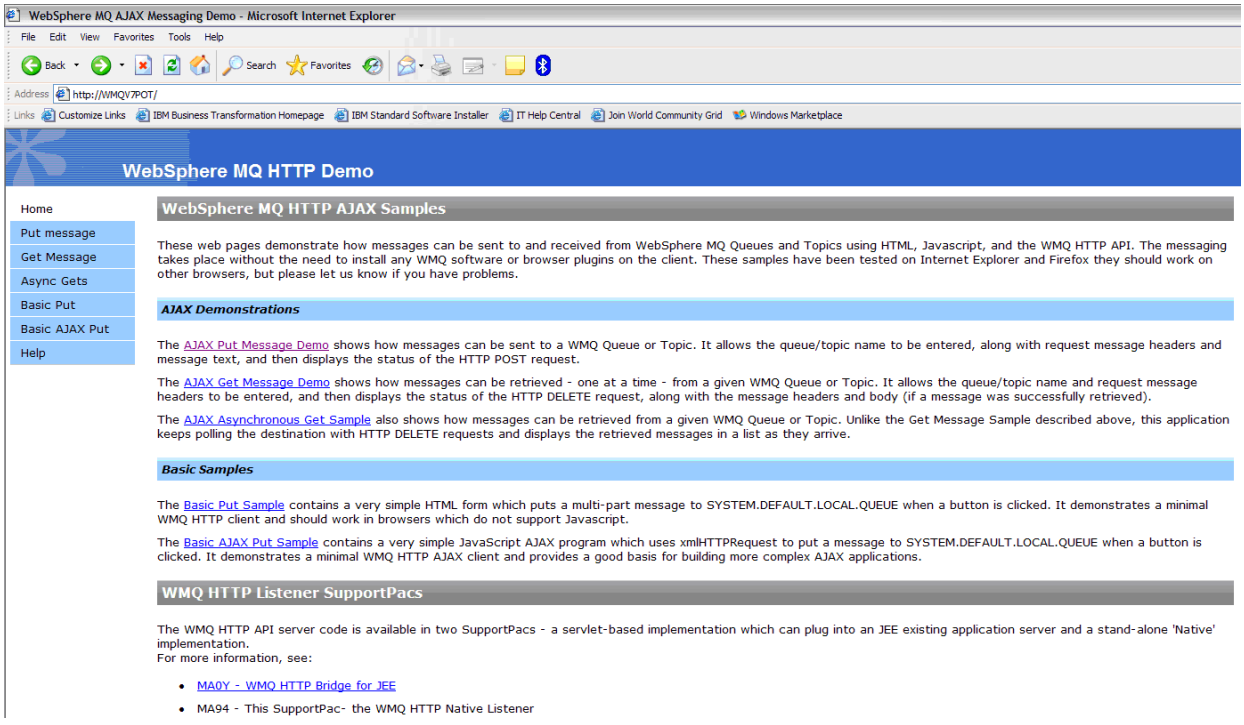


If you have any difficulty switching to the native desktop, ask the instructor for assistance.

__12. **From the native desktop of your machine, open** a web **browser** and enter **http://WMQV7POT** as the url and hit **Enter**

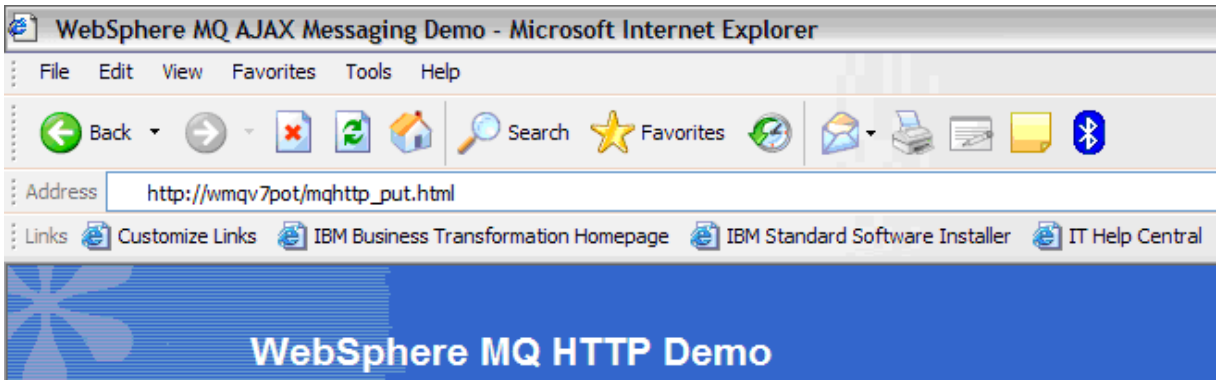


__13. Verify that the following welcome screen is displayed.

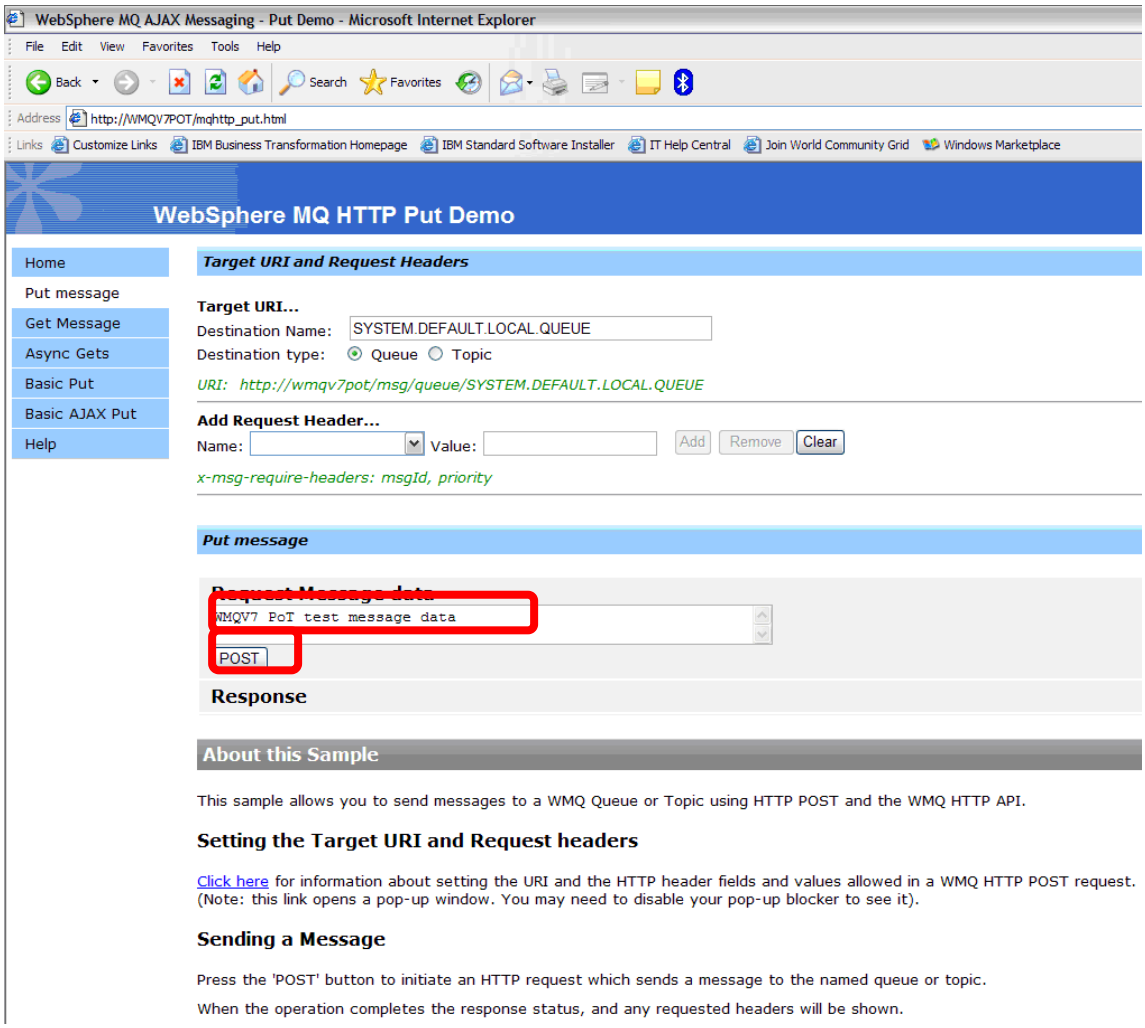


5.10 Using the PUTQ JavaScript

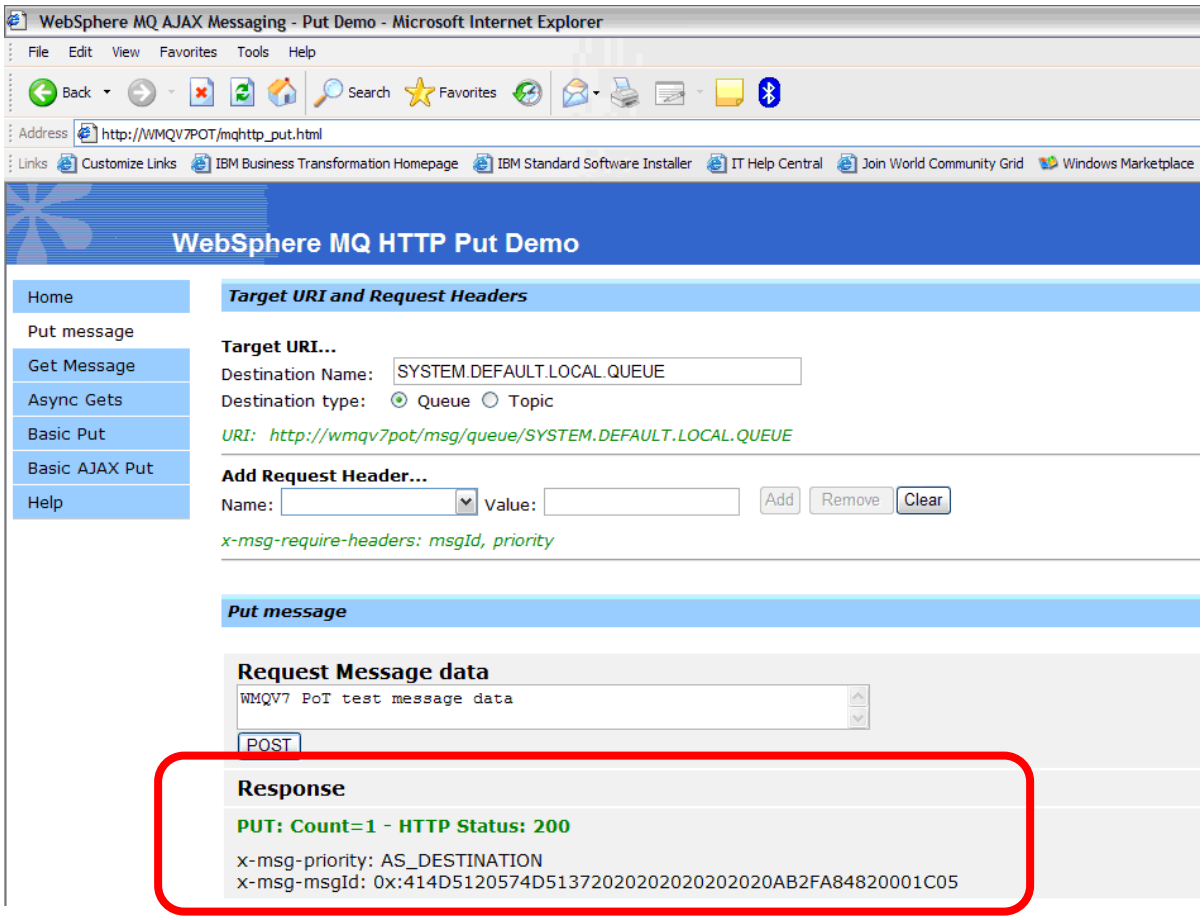
__1. Enter the url http://wmqv7pot/mqhttp_put.html and hit Enter



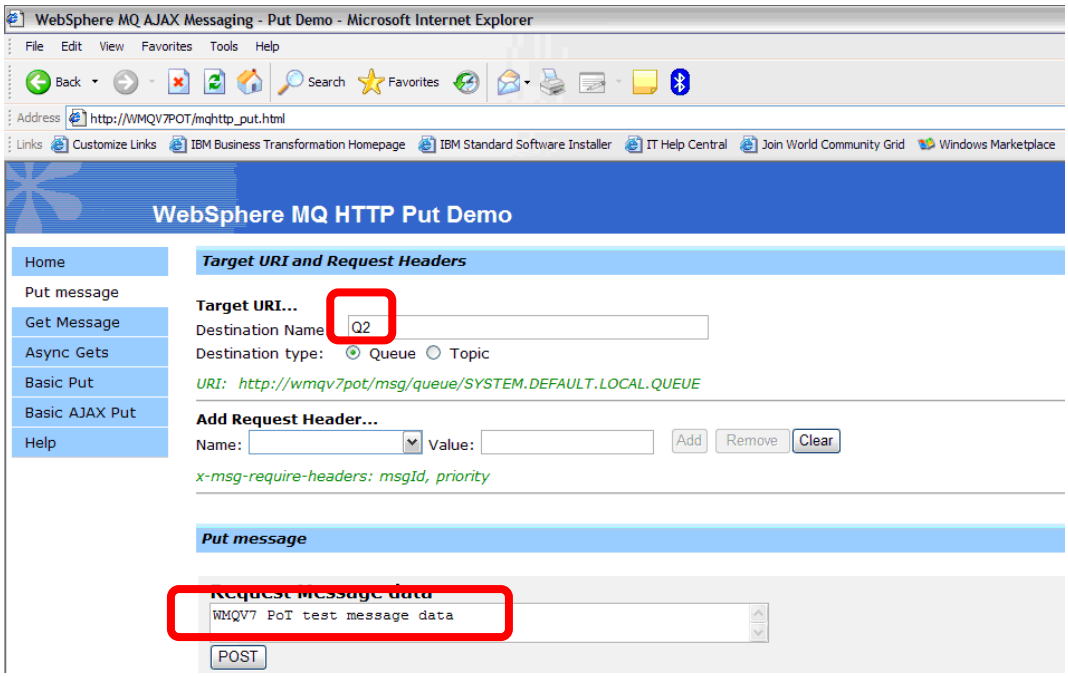
__2. Enter "**WMQV7 PoT test message data**" in the **Request Message data** box and press the **POST** button.



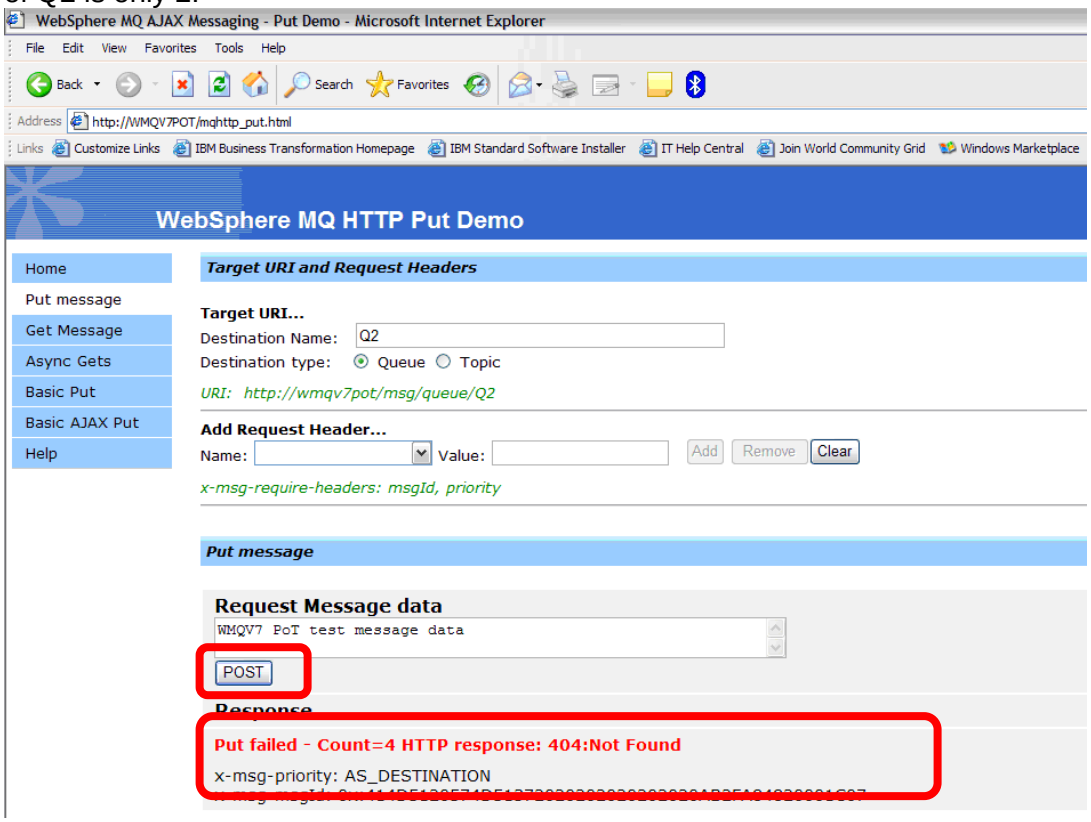
__3. Observe that you get an **HTTP** 200 response as well as the priority and **msgid** of the message you successfully placed on the queue.



__4. Change the Destination Name to Q2 and enter “WMQV7 PoT test message data” in the Request Message data box.



__5. Press the **POST** button **three times**. The third time you **should get an error** – The max depth of Q2 is only 2.



This concludes Lab 5.

Appendix A. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have

been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental. All references to fictitious companies or individuals are used for illustration purposes only.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Appendix B. Trademarks and copyrights

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	AIX	CICS	ClearCase	ClearQuest	Cloudscape
Cube Views	DB2	developerWorks	DRDA	IMS	IMS/ESA
Informix	Lotus	Lotus Workflow	MQSeries	OmniFind	
Rational	Redbooks	Red Brick	RequisitePro	System i	
System z	*Tivoli*	*WebSphere*	*Workplace*	*System p*	

Adobe, Acrobat, Portable Document Format (PDF), and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. See Java Guidelines

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

ITIL is a registered trademark and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Other company, product and service names may be trademarks or service marks of others.

NOTES

NOTES

Please Recycle

IBM Software