Enabling MQ classes for Java and classes for JMS trace dynamically

Paul_Titheridge
Published on 27/10/2017

I've been working with a customer recently who wanted to be able to turn on MQ classes for JMS trace dynamically, and then stop the trace when they noticed that their application had reported a specific JMSException. In the past, there was no way to do this. Trace could only be enabled when an application started up, and would be stopped when the application finished.

However, MQ V8 and V9 ship a utility called traceControl, which provides the ability to turn MQ classes for Java and classes for JMS trace on and off while an application is running. The utility interacts with a Java Managed Bean (MBean) provided by the MQ messaging client trace mechanism to control trace.

In this blog post, we will look at some examples of how to use the utility.

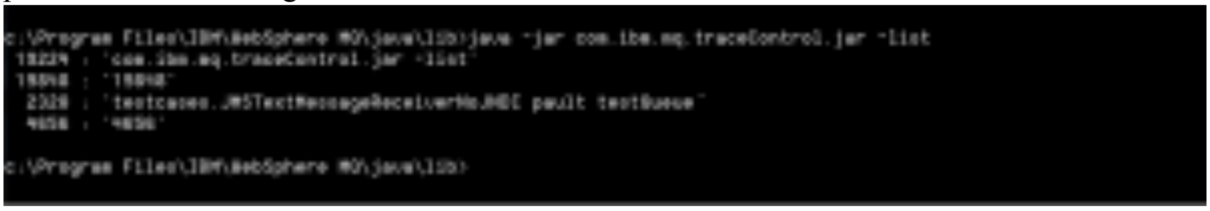
Identifying the process to trace

On my test system, I have an MQ classes for JMS application called JMSTextMessageReceiverNoJNDI, which gets messages from a queue destination in a loop. Let's suppose that I want to trace this application.

The first thing I have to do is to identify the process identifier for the application. I do this by bringing up a command prompt, navigating to the directory MQ_INSTALLATION_PATH/java/lib and running the following command:

```
java -jar com.ibm.mq.traceControl.jar -list
```

The traceControl utility will now look for all the Java processes on the local system. For every Java process that it finds, it will display the process identifier and the arguments that were passed into the java command used to start that process. On my test system, this produces the following results.



```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -list
 19224 : 'com.ibm.mq.traceControl.jar -list'
 19848 : '19848'
  2328 : 'testcases.JMSTextMessageReceiverNoJNDI pault testQueue'
  4656 : '4656'

c:\Program Files\IBM\WebSphere MQ\java\lib>
```

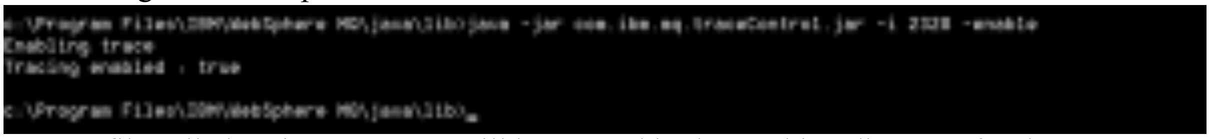
The Java process for my application is the third one, and the process identifier is 2328 .

Turning on trace

Now that I have identified the process identifier for my application, I can turn on MQ classes for JMS trace for it by running the command:

```
java -jar com.ibm.mq.traceControl.jar -i 2328 -enable
```

This causes the traceControl utility to connect to the Java process, and invoke the MBean provided by the MQ messaging client trace mechanism to turn on trace. Running the command gives the output shown below:

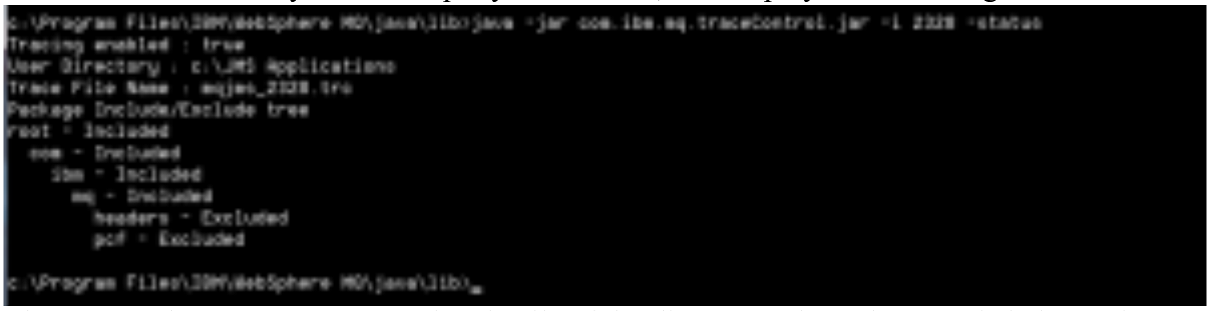


```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -i 2328 -enable
Enabling trace
Tracing enabled : true

c:\Program Files\IBM\WebSphere MQ\java\lib>_
```

A trace file called mqjms_2328.trc will be created in the working directory for the application. To confirm what the working directory is, I can run the command:

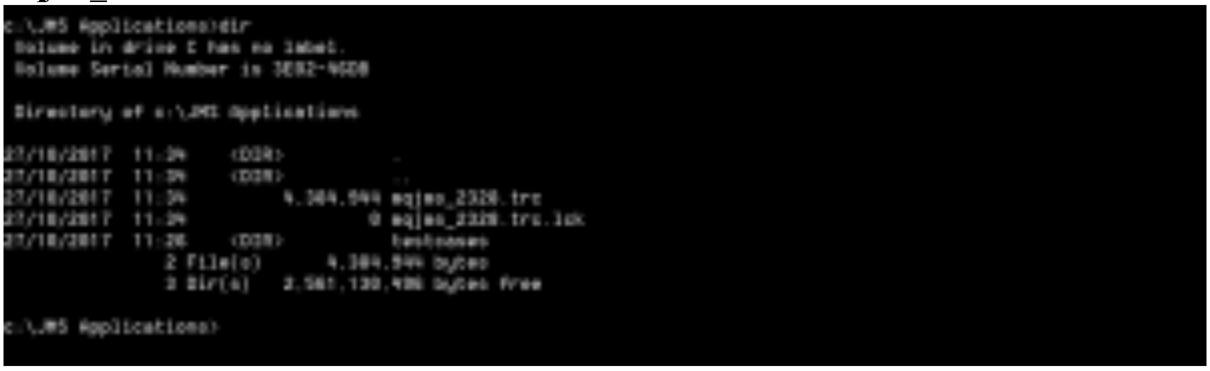
```
java -jar com.ibm.mq.traceControl.jar -i 2328 -status
```

The traceControl utility will now query the MBean, and display the following information:



```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -i 2328 -status
Tracing enabled : true
User Directory : c:\JMS Applications
Trace File Name : mqjms_2328.trc
Package Include/Exclude tree
root - Included
  com - Included
    ibm - Included
      mq - Included
        headers - Excluded
        pcf - Excluded

c:\Program Files\IBM\WebSphere MQ\java\lib>_
```

The “User Directory” entry contains details of the directory where the trace is being written – in my case, this is “C:\JMS Applications”. If I look in this directory, I should see a file called mqjms_2328.trc:



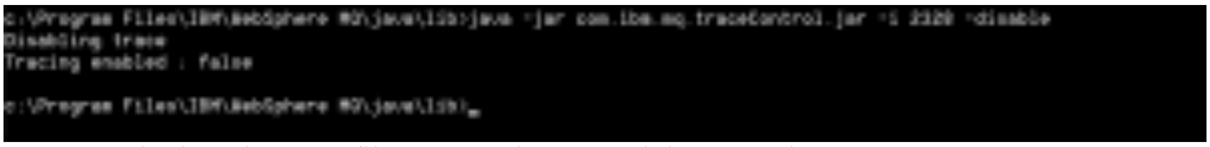
```
c:\JMS Applications>dir
 Volume in drive C has no label.
 Volume Serial Number is 3E82-4GD8

 Directory of c:\JMS Applications

27/10/2017  11:34    <DIR>          .
27/10/2017  11:34    <DIR>          ..
27/10/2017  11:34         4,384,944 mqjms_2328.trc
27/10/2017  11:34                 0 mqjms_2328.trc.lck
27/10/2017  11:26    <DIR>          testcases
               2 File(s)      4,384,944 bytes
               3 Dir(s)   2,581,130,496 bytes free

c:\JMS Applications>
```

Turning trace off

After running for a while, I decide to turn off trace (maybe the application has reported the exception that I am interested in). To do this, I run this command:

```
java -jar com.ibm.mq.traceControl.jar -i 2328 -disable
```

The traceControl utility will connect to the MBean, turn trace off and display the following information:



```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -i 2328 -disable
Disabling trace
Tracing enabled : false

c:\Program Files\IBM\WebSphere MQ\java\lib>_
```

I can now look at the trace file, to see what caused the exception to occur.

Configuring trace

One nice thing about the traceControl utility is that works in conjunction with the MQ classes for Java configuration file and the MQ classes for JMS configuration file. These configuration files are used to configure the MQ messaging client trace mechanism, and allow you to specify things such as:

- The name and location of the trace file that will be created.
- The maximum size of the trace file.
- The number of trace files to use to produce wrapping trace.

If your application has been configured to use a configuration file, then the traceControl utility will use the information in that file when writing trace information.

For example, suppose that I have created an MQ classes for JMS configuration file called C:\mqjms.config, which contains the following entries:

```
com.ibm.msg.client.commonservices.trace.status=OFF
com.ibm.msg.client.commonservices.trace.outputName=C:/Trace/MQJMSTrace.trc
com.ibm.msg.client.commonservices.trace.limit=10000000
com.ibm.msg.client.commonservices.trace.count=5
```

These properties will cause the MQ messaging client trace mechanism to use up to 5 trace files, called:

- MQJMSTrace.trc
- MQJMSTrace.trc.1
- MQJMSTrace.trc.2
- MQJMSTrace.trc.3
- MQJMSTrace.trc.4

in the directory C:\Trace, to store trace information, where the maximum size of each trace file will be approximately 10000000 bytes (10MB). The first line in the configuration file is important, as it means that trace will not be enabled when the application starts up.

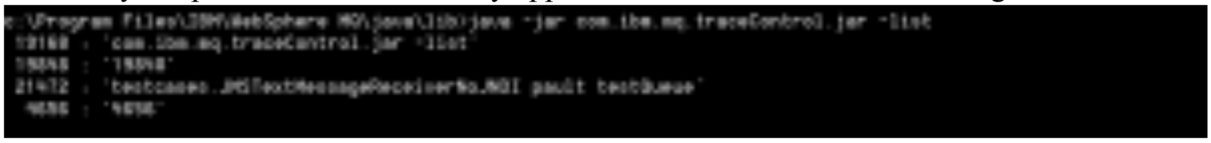
Now, I run my application, specifying the Java system property so that the MQ classes for JMS load the MQ classes for JMS configuration file:

```
java -Dcom.ibm.msg.client.config.location=file:/C:/mqjms.config
testcases.JMSTextMessageReceiverNoJNDI pault testQueue
```

When the application starts up, trace is not enabled and so no trace files are generated. Next, I use the command:

```
java -jar com.ibm.mq.traceControl.jar -list
```

to identify the process identifier for my application. This returns the following information:

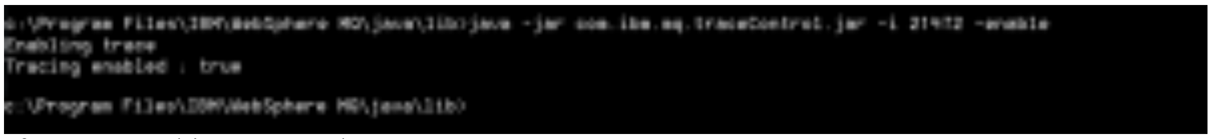


```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -list
 19168 : 'com.ibm.mq.traceControl.jar -list'
 19848 : '19848'
 21472 : 'testcases.JMSTextMessageReceiverNoJNDI pault testQueue'
  4656 : '4656'
```

The process identifier for my application is 21472, so I run the command:

```
java -jar com.ibm.mq.traceControl.jar -i 21472 -enable
```

to enable trace. The command returns the information shown below:

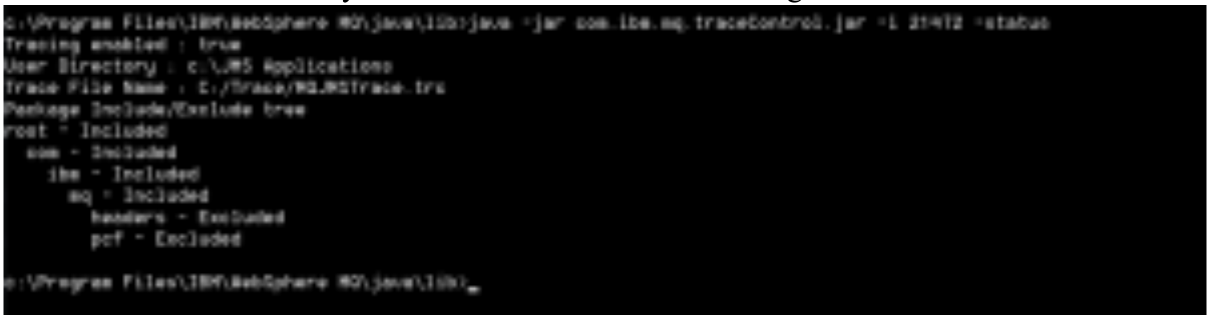


```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -i 21472 -enable
Enabling trace
Tracing enabled : true

c:\Program Files\IBM\WebSphere MQ\java\lib>
```

If I now run this command:

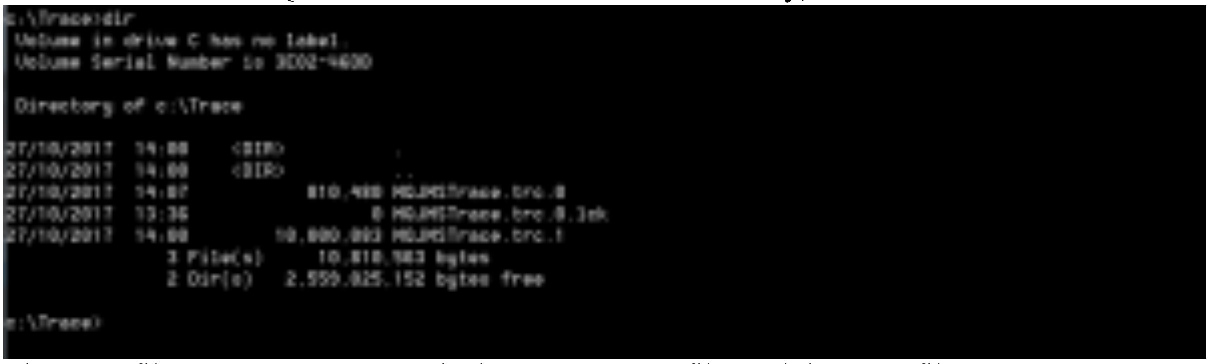
```
java -jar com.ibm.mq.traceControl.jar -i 21472 -status
```

then the traceControl utility will show where the trace is being written to:



```
c:\Program Files\IBM\WebSphere MQ\java\lib>java -jar com.ibm.mq.traceControl.jar -i 21412 -status
Tracing enabled : true
User Directory : c:\JMS Applications
Trace File Name : C:/Trace/MQJMSTrace.trc
Package Include/Exclude tree
root - Included
  com - Included
    ibm - Included
      mq - Included
        headers - Excluded
        pcf - Excluded

c:\Program Files\IBM\WebSphere MQ\java\lib>_
```

The “Trace File Name” entry here is fully qualified, and shows that the trace is being written to the file C:\trace\MQJMSTrace.trc. If I look in this directory, I see two trace files:



```
c:\Trace>dir
 Volume in drive C has no label.
 Volume Serial Number is 3C02-4630

 Directory of c:\Trace

27/10/2017  14:08    <DIR>          .
27/10/2017  14:08    <DIR>          ..
27/10/2017  14:07           810,480 MQJMSTrace.trc.0
27/10/2017  13:36                 0 MQJMSTrace.trc.0.lck
27/10/2017  14:08        10,000,003 MQJMSTrace.trc.1
               3 File(s)     10,810,483 bytes
               2 Dir(s)   2,559,025,152 bytes free

c:\Trace>
```

The trace file MQJMSTrace.trc.0 is the current trace file, and the trace file MQJMSTrace.trc.1 contains historical trace data. The MQ classes for JMS initially started writing to a file called MQJMSTrace.trc.0. When this file reached its maximum size of 10MB, it was renamed to MQJMSTrace.trc.1 and a new trace file called MQJMSTrace.trc.0 was created to store the current trace data.

Dynamically enabling trace when using non-IBM JVMs

My test system was using an IBM Java Runtime Environment. If you have applications that run using a non-IBM Java Runtime Environment, then you need to ensure that the JAR file:
`<JAVA_HOME>/lib/tools.jar`
provided with the Java Runtime Environment is on the classpath for the traceControl utility. This file contains some classes that are used by the utility to connect to the Java process. If the utility is run using a classpath that does not contain the JAR file, then it will report the following exception:

```
Unable to work in this JVM Implementation
java.lang.ClassNotFoundException: com.sun.tools.attach.VirtualMachine
    at java.net.URLClassLoader.findClass(Unknown Source)
    at java.lang.ClassLoader.loadClass(Unknown Source)
```

Dynamically enabling trace for applications running in WebSphere Application Server

One final thing to note is that the traceControl utility should not be used to dynamically enable or disable trace for applications running inside of either WebSphere Application Server Traditional or WebSphere Application Server Liberty:

- For WebSphere Application Server Traditional, trace can be controlled dynamically via the WebSphere Administrative Console. Details of the trace specification to use can be found here.
- When using WebSphere Application Server Liberty, trace is enabled by adding the entry:
  ```
  <logging
  traceSpecification="JMSApi=all:Messaging=all:com.ibm.mq.*=all:Transac
  tion=all:"/>
  ```
  to the server.xml file. If the file is updated when the application server is running, then WebSphere Application Server Liberty will detect that the file has changed and then start tracing.

Further Reading

And that's it! Hopefully this has given you an insight into how the traceControl utility works and how it can be used to collect trace from running applications. If you want more information about the utility, then take a look at the topic called "Controlling trace in a running process by using IBM MQ classes for Java and IBM MQ classes for JMS" in the MQ V8 and V9 sections of IBM Knowledge Center.

As always, I hope this helps and if you have any questions, feel free to ask.

Tags java, jms, mq, trace

by Paul_Titheridge