ENSEMBLE TO RABBITMQ

JAVA CLIENT QUICK START GUIDE

SOFTWARE VERSIONS

SERVER

OS	Windows Server 2012 R2 x64
Erlang	OTP 19.1
RabbitMQ Server	3.6.5
RabbitMQ Java Client	3.6.5

CLIENT

OS	Windows Server 2012 R2 x64
Java	JDK SE 8u111 JRE SE 8u111
RabbitMQ Java Client	3.6.5
Ensemble	2016.2.0.736.0

RABBITMQ

INSTALL SERVER

Download and run the Erlang Windows Binary File.
Download and install the RabbitMQ Server.

http://www.rabbitmq.com/install-windows.html

OPEN PORT 5672 ON SERVER FIREWALL

5672 is the default non-SSL port the RabbitMQ server listens for AMQP connections.

If not already configured during the RabbitMQ Server installation then open the Windows Firewall with Advanced Security management console:

Control Panel > Administrative Tools > Windows Firewall with Advanced Security

Add a new Port Inbound Rule:

Protocol	TCP
Port	5672

And any other ports required for this installation.

INSTALL CLIENT

Install client on both the server hosting the RabbitMQ Server and a machine acting as a remote client.

Download and install the Java JDK.

Set and check JAVA_HOME, example:

System > Advanced system settings > Environment Variables…

Variable name: **JAVA_HOME**

Variable value: **C:\Program Files\Java\jdk1.8.0_111**

C:\>echo %JAVA_HOME%

C:\Program Files\Java\jdk1.8.0_111

Add the JDK bin folder to the system PATH, example:

System > Advanced system settings > Environment Variables…

Variable name: Path

Variable value: %SystemRoot%;%SystemRoot%\system32;
%SystemRoot%\System32\Wbem;%SYSTEMROOT%\System32\WindowsPowerShell
\v1.0\;**C:\Program Files\Java\jdk1.8.0_111\bin**

Download, unzip and copy the RabbitMQ Java Client files to a folder:

C:\rabbitmq-java-client-bin-3.6.5\commons-cli-1.1.jar
C:\rabbitmq-java-client-bin-3.6.5\commons-io-1.2.jar
C:\rabbitmq-java-client-bin-3.6.5\junit.jar
…
C:\rabbitmq-java-client-bin-3.6.5\stresspersister.sh

RPC LOCAL QUICK TEST

Local to RabbitMQ Server to facilitate confidence testing.

Download RabbitMQ Performance Testing Tool jar file:
http://central.maven.org/maven2/com/rabbitmq/perf-test/1.0.1/perf-test-1.0.1.jar
Copy the jar file to the RabbitMQ Java Client folder:
C:\rabbitmq-java-client-bin-3.6.5\perf-test-1.0.1.jar

More details available at:
https://www.rabbitmq.com/java-tools.html
https://github.com/rabbitmq/rabbitmq-perf-test

NOTES:

The following are defaulted to:

host name	"amqp://localhost"
user name	“guest”
Password	“guest”
port number	5672
Vhost	“/”

RUN HELLOSERVER EXAMPLE

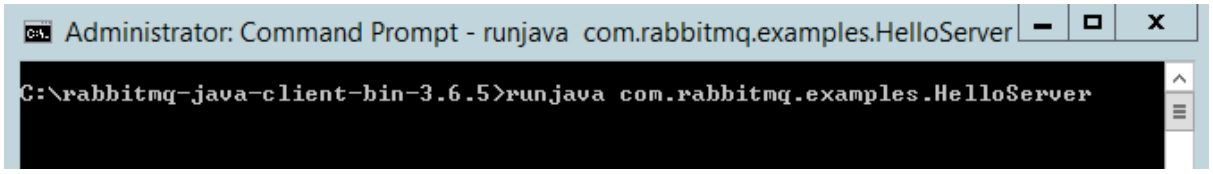
Open a command prompt, navigate to the RabbitMQ Java client folder and execute:

RabbitMQ Java client path>runjava com.rabbitmq.examples.HelloServer

Example:

Windows:

C:\rabbitmq-java-client-bin-3.6.5>runjava com.rabbitmq.examples.HelloServer



Unix:
$ sh runjava.sh com.rabbitmq.examples.HelloServer

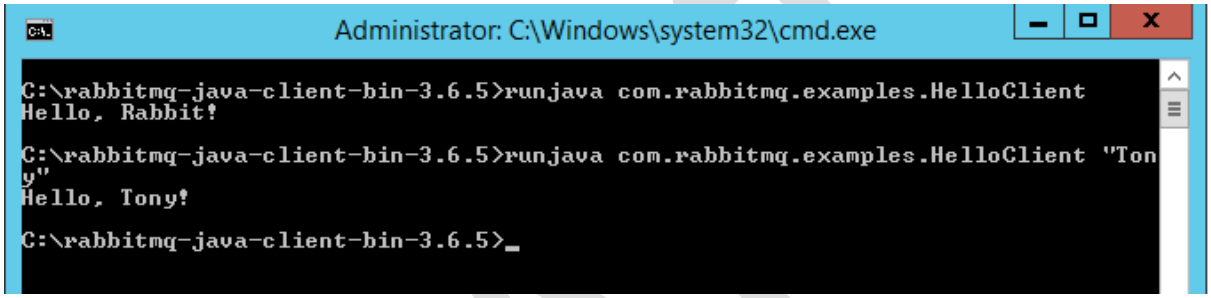
RUN HELLOCLIENT EXAMPLE

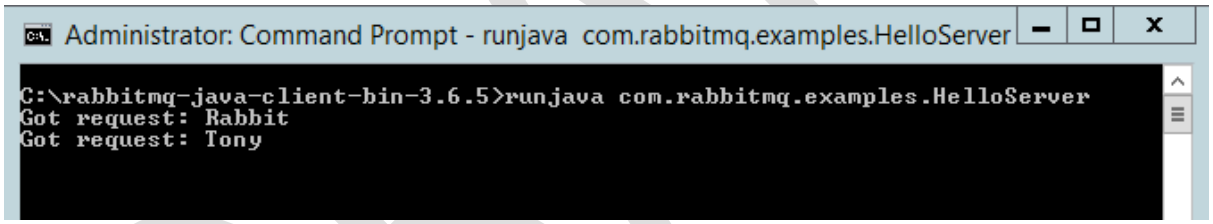
Open a separate command prompt, navigate to the Java client folder and execute:

Java client path>runjava com.rabbitmq.examples.HelloClient

Example:

C:\rabbitmq-java-client-bin-3.6.5>runjava com.rabbitmq.examples.HelloClient
C:\rabbitmq-java-client-bin-3.6.5>runjava com.rabbitmq.examples.HelloClient "Tony"





Unix:
$ sh runjava.sh com.rabbitmq.examples.HelloClient
Hello, Rabbit!
$ sh runjava.sh com.rabbitmq.examples.HelloClient "Tony"
Hello, Tony!

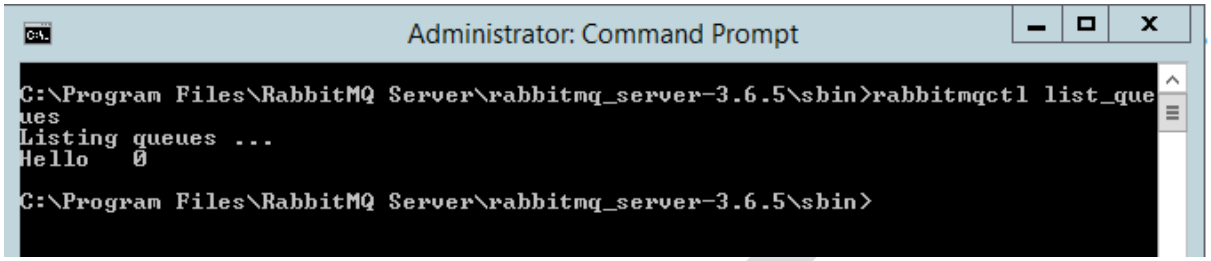
MONITOR QUEUES

Open a separate command prompt, navigate to the *sbin* folder in the RabbitMQ server folder and execute:

RabbitMQ Server path>\sbin\rabbitmqctl list_queues

Example:

C:\Program Files\RabbitMQ Server\rabbitmq_server-3.6.5\sbin>rabbitmqctl list_queues



Unix:
$ ./rabbitmqctl list_queues
Listing queues ...
Hello 0

RPC REMOTE QUICK TEST

Remote to RabbitMQ server to facilitate confidence testing across network.

NOTES:

The following are defaulted to:

user name	"guest"
Password	"guest"
port number	5672
Vhost	"/"

"guest" user can only connect via localhost:
https://www.rabbitmq.com/access-control.html

RUN HELLOSERVER EXAMPLE

On the remote machine; open a command prompt, navigate to the RabbitMQ Java client folder and execute:

RabbitMQ Java client path>runjava com.rabbitmq.examples.HelloServer "*host*"

Example:

Windows:
C:\rabbitmq-java-client-bin-3.6.5>runjava com.rabbitmq.examples.HelloServer "amqp://WIN-UK0KESAK7LB"

Unix:
$ sh runjava.sh com.rabbitmq.examples.HelloServer

RUN HELLOCLIENT EXAMPLE

On the remote machine; open a separate command prompt, navigate to the Java client folder and execute:

Java client path>runjava com.rabbitmq.examples.HelloClient "*message*" "*host*"

Example:

Windows:

```
C:\rabbitmq-java-client-bin-3.6.5>runjava com.rabbitmq.examples.HelloClient "Tony" "amqp://WIN-UK0KESAK7LB"
```

Unix:

```
$ sh runjava.sh com.rabbitmq.examples.HelloClient "Tony" "amqp://WIN-UK0KESAK7LB"
Hello, Tony!
```

MONITOR QUEUES

On the RabbitMQ Server server; open a separate command prompt, navigate to the *sbin* folder in the RabbitMQ server folder and execute:

RabbitMQ Server path>\sbin\rabbitmqctl list_queues

Example:

Windows:

```
C:\Program Files\RabbitMQ Server\rabbitmq_server-3.6.5\sbin>rabbitmqctl list_queues
```

Unix:

```
$ ./rabbitmqctl list_queues
Listing queues ...
Hello      0
```

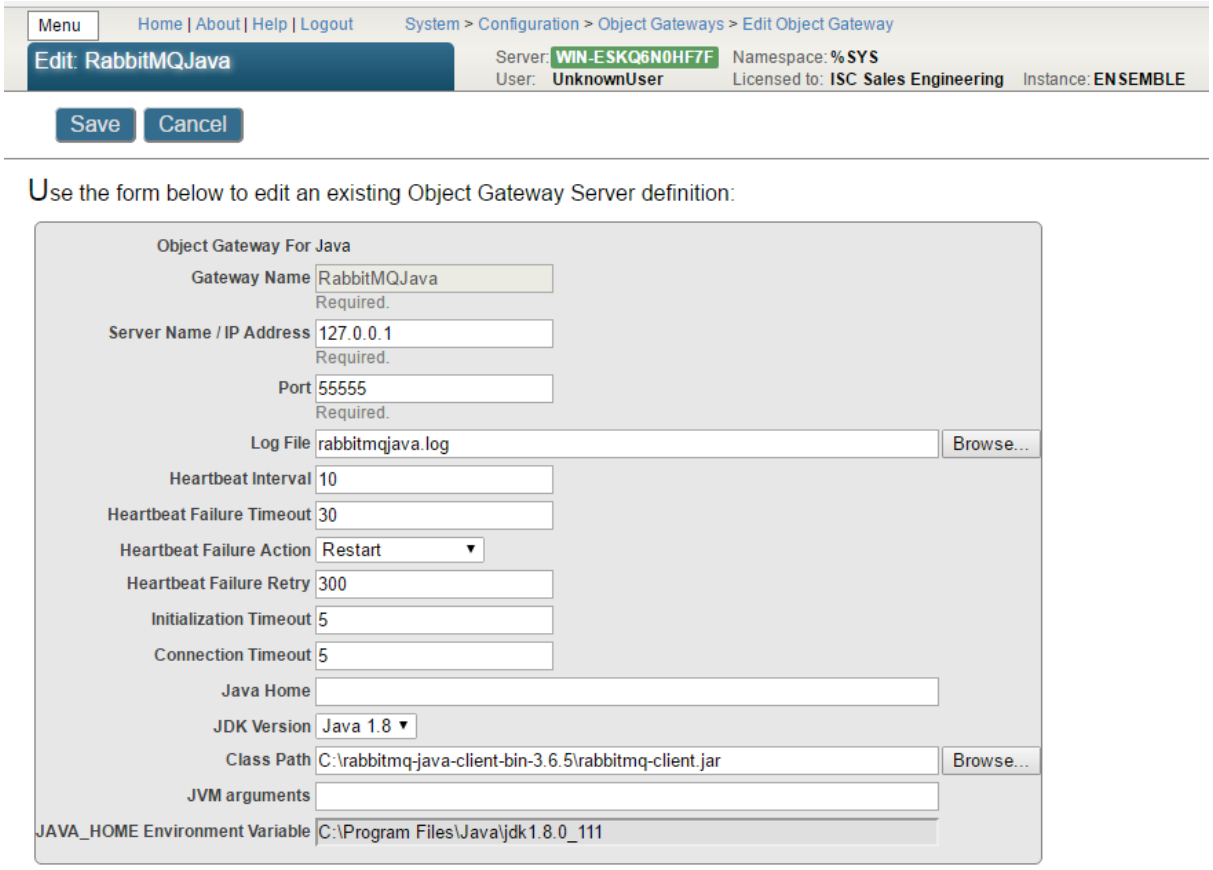
ENSEMBLE OBJECT GATEWAY

CREATE JAVA OBJECT GATEWAY DEFINITION

Navigate to the Object Gateways page in the Management Portal:

System > Configuration > Connectivity > Object Gateways

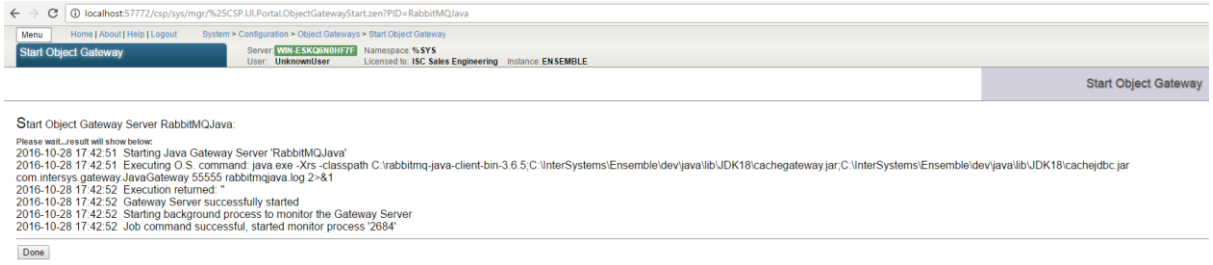
and select Create New Gateway.



START INSTANCE OF GATEWAY

You can start the Java Gateway server in one of the following ways:

- Manually, by selecting the Start link of a previously configured gateway (see below)
- Manually, by calling the business service StartGateway method
- Manually, by entering a command at the Terminal command prompt
- Automatically, by adding a Java Gateway business service to the production
    - The Java Gateway server starts when the production starts



PROXY CLASS

The object gateway provides a proxy class mechanism to execute, in this example, Java code, from within Ensemble.

RABBITMQ JAVA WRAPPER CLASS

It is usually not practical to import a complete library, so the recommendation is to create a wrapper class that provides a simplified, subset of the required functionality.

CREATE A JAVA SOURCE FILE

Example: *<root folder>\com\myorgname\rabbitmq\Wrapper.java*

```
package com.myorgname.rabbitmq;

import com.rabbitmq.client.ConnectionFactory;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.QueueingConsumer;


public class Wrapper {

	public void sendMsg(String hostName, String queueName, byte[] msg) throws Exception {
		ConnectionFactory factory = new ConnectionFactory();
		factory.setHost(hostName);
		Connection connection = factory.newConnection();
		Channel channel = connection.createChannel();
		channel.queueDeclare(queueName, false, false, false, null);

		channel.basicPublish("", queueName, null, msg);

		channel.close();
		connection.close();

	}

	public int readMsg(String hostName, String queueName, byte[] msg) throws Exception {
		ConnectionFactory factory = new ConnectionFactory();
		factory.setHost(hostName);
		Connection connection = factory.newConnection();
		Channel channel = connection.createChannel();
		channel.queueDeclare(queueName, false, false, false, null);
		QueueingConsumer consumer = new QueueingConsumer(channel);
		channel.basicConsume(queueName, true, consumer);

		QueueingConsumer.Delivery delivery = consumer.nextDelivery();
		int len = delivery.getBody().length;
		System.arraycopy(delivery.getBody(),0,msg,0,len);

		channel.close();
		connection.close();

		return len;

	}

}
```

COMPILE THE WRAPPER CLASS

Compile the class using for example:

javac -verbose -cp C:\rabbitmq-java-client-bin-3.6.5\rabbitmq-client.jar com\myorgname\rabbitmq\Wrapper.java

Example output:

```
C:\rabbitmq-java-proxy-3.6.5>javac -verbose -cp C:\rabbitmq-java-client-bin-3.6.
5\rabbitmq-client.jar com\myorgname\rabbitmq\Wrapper.java
[parsing started RegularFileObject[com\myorgname\rabbitmq\Wrapper.java]]
[parsing completed 30ms]
[search path for source files: C:\rabbitmq-java-client-bin-3.6.5\rabbitmq-client
.jar]
…
[loading ZipFileIndexFileObject[C:\rabbitmq-java-client-bin-3.6.5\rabbitmq-clien
t.jar(com/rabbitmq/client/AMQP$BasicProperties.class)]]
[wrote RegularFileObject[com\myorgname\rabbitmq\Wrapper.class]]
[total 631ms]
```

PACKAGE THE CLASS/ES IN A JAR FILE

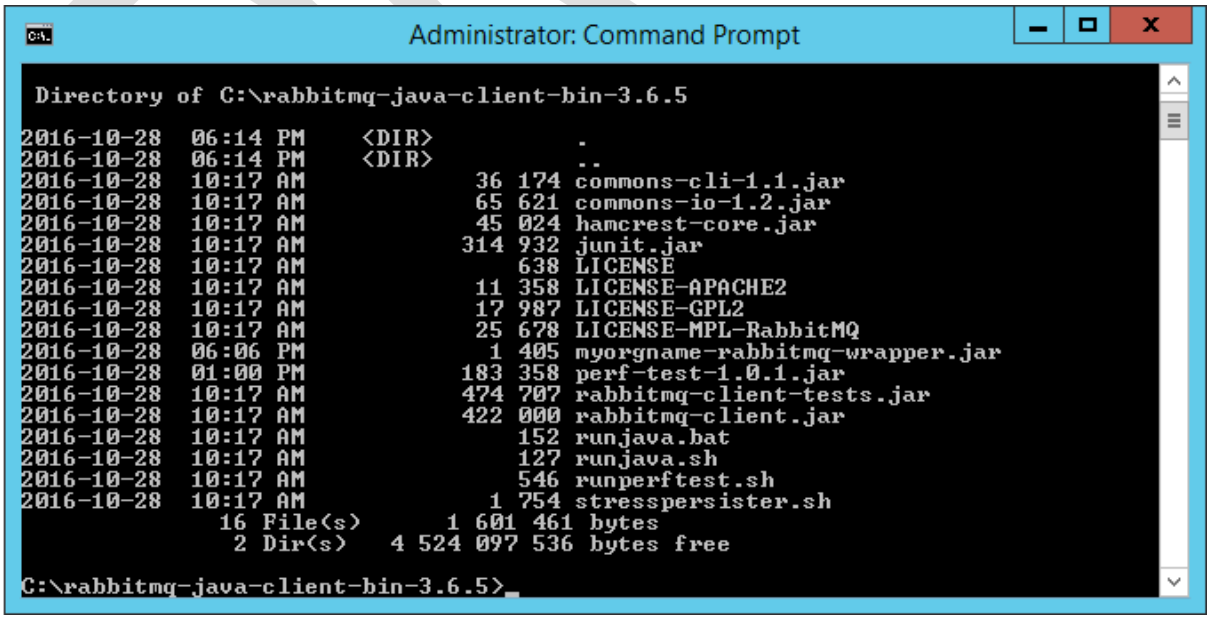
Create a jar using for example:

jar cvf myorgname-rabbitmq-wrapper.jar com\myorgname\rabbitmq\Wrapper.class

Example output:

```
C:\rabbitmq-java-proxy-3.6.5>jar cvf myorgname-rabbitmq-wrapper.jar com\myorgnam
e\rabbitmq\Wrapper.class
added manifest
adding: com/myorgname/rabbitmq/Wrapper.class(in = 1938) (out= 899)(deflated 53%)
```

COPY THE JAR FILE TO THE RABBITMQ JAVA CLIENT INSTALL FOLDER

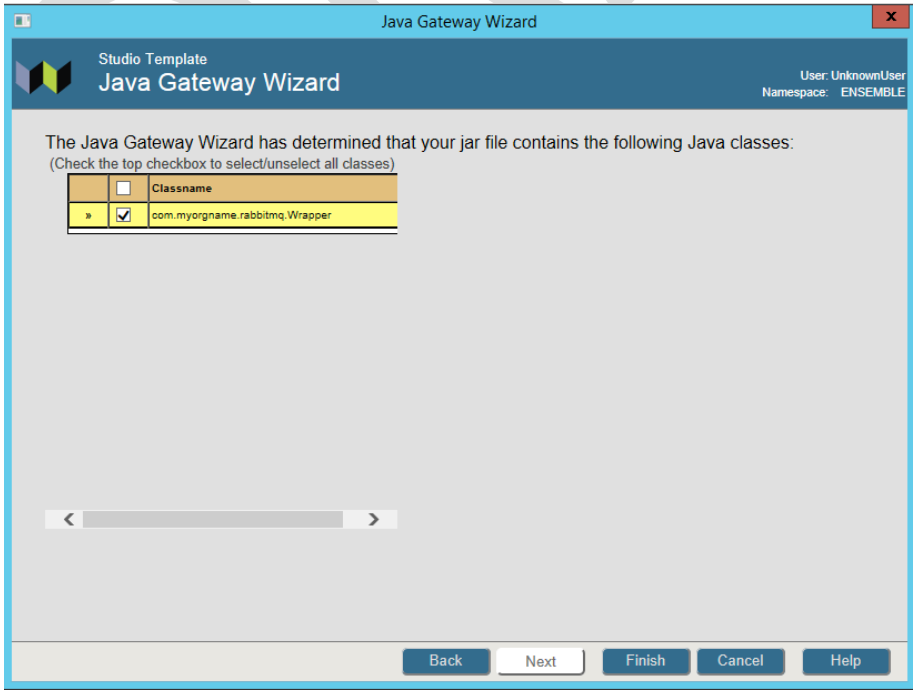


ENSEMBLE PROXY CLASS

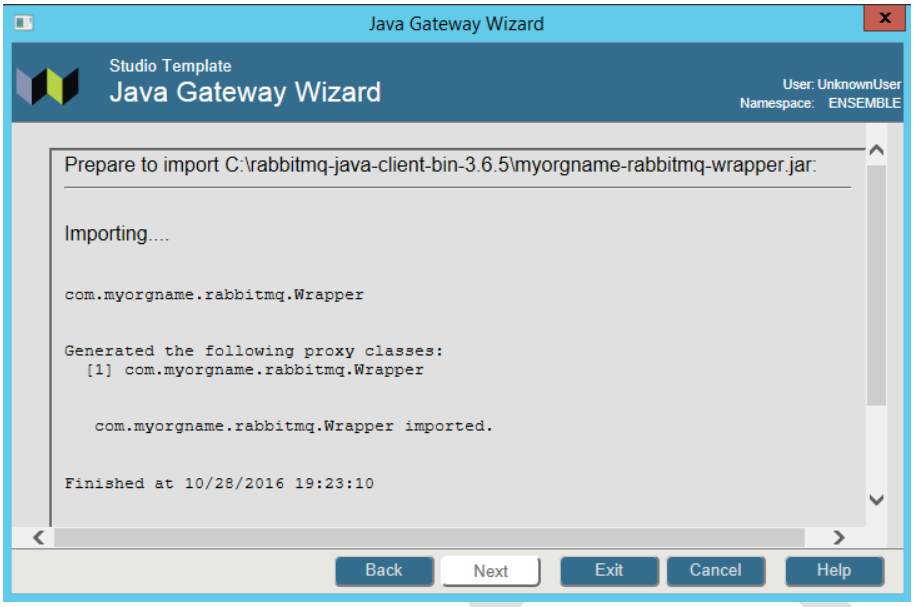
IMPORT THE WRAPPER INTO ENSEMBLE

- Open the Java Gateway Wizard in Studio:
  Tools > Add-Ins > Java Gateway Wizard
- Select Jar File and enter the path and name of the wrapper jar file
- Specify the Java Gateway server name or IP address and its port number
- Select Next

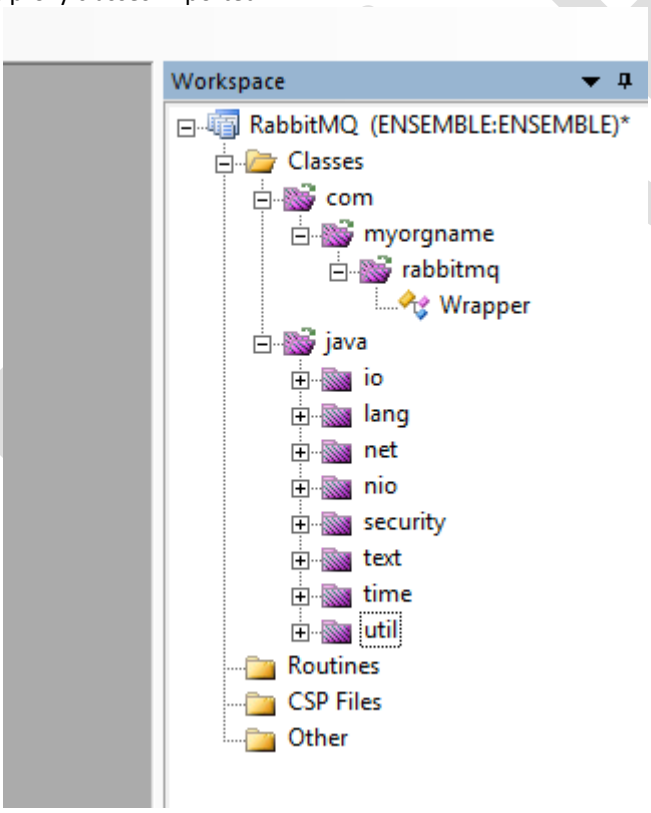
- Select the Wrapper class
- Select Finish



Example result of an import:



Note the proxy classes imported:



USE THE PROXY CLASSES IN YOUR APPLICATION

CREATE A UTILITY CLASS TO TEST CONNECTIVITY TO THE RABBITMQ SERVER

```
Class RabbitMQ.Java.HelloWorld Extends %RegisteredObject
{

Parameter CLASSPATH = "C:\rabbitmq-java-client-bin-3.6.5\myorgname-rabbitmq-wrapper.jar";

Parameter HOST = "localhost";

Parameter QUEUE = "hello";

/// s sc=##class(RabbitMQ.Java.HelloWorld).SendMsg()
ClassMethod SendMsg(pMsg = "Hello from Ensemble!") As %Status
{
	#dim tGateway as %Net.Remote.Gateway
	#dim tException as %Exception.AbstractException

	Set tSC=$$$OK
	Try	{
		Set tGateway=..Connect()
		Set tRabbitMQWrapper=##class(com.myorgname.rabbitmq.Wrapper).%New(tGateway)
		//*** Do not use this syntax. It does not work! ***
		//		Set tByteStream=##class(%Library.GlobalBinaryStream).%New("Hello
World!")
		// ***********************************************
		Set tByteStream=##class(%Library.GlobalBinaryStream).%New()
		Set tSC = tByteStream.Write(pMsg)
		Do tRabbitMQWrapper.sendMsg(..#HOST,..#QUEUE, tByteStream)
		Write !,"Sent message via "_tByteStream.Read()

		Set tSC=tGateway.%Disconnect()

	} Catch tException {
		Set tSC = tException.AsStatus()
	}

	Quit tSC
}

/// s sc=##class(RabbitMQ.Java.HelloWorld).ReadMsg()
ClassMethod ReadMsg(pMsgLen = 32000) As %Status
{
	#dim tGateway as %Net.Remote.Gateway
	#dim tException as %Exception.AbstractException

	Set tSC=$$$OK
	Try	{
		Set tGateway=..Connect()
		Set tRabbitMQWrapper=##class(com.myorgname.rabbitmq.Wrapper).%New(tGateway)

		Set tReadStream=##class(%GlobalBinaryStream).%New()
		// we need to 'reserve' a number of bytes since we are passing the stream
		// by reference (Java's equivalent is byte[] ba = new byte[max];)
		For i=1:1:pMsgLen Do tReadStream.Write("0")
```

```
		Set tBytesRead=tRabbitMQWrapper.readMsg(..#HOST,..#QUEUE, .tReadStream)
		Write tReadStream.Read(tBytesRead),!
		Write "Bytes Read: ",tBytesRead,!

		Set tSC=tGateway.%Disconnect()

	} Catch tException {
		Set tSC = tException.AsStatus()
	}

	Quit tSC
}

ClassMethod Connect(pPort As %Integer = 55555, pHost As %String = "127.0.0.1") As %Net.Remote.Gateway
{
	// connect to current namespace, use 2 second timeout
	Set tSC=$$$OK,tNamespace=$zu(5),tTimeout=2
	Set tClassPath=##class(%ListOfDataTypes).%New()
	Do tClassPath.Insert(..#CLASSPATH)

	// get a connection handle and connect
	Set tGateway=##class(%Net.Remote.Gateway).%New()
	Set tSC=tGateway.%Connect(pHost,pPort,tNamespace,tTimeout,tClassPath)
	If tSC'=$$$OK {
		Write $system.OBJ.DisplayError(tSC)
		Set tGateway=""
	}
	Quit tGateway
}
```

TEST THE UTILITY CLASS