Simple JMS 1.1 Producer and Consumer Example With Eclipse and Embedded JBoss HornetQ Server

Before going to through this post, please read my previous post at "JMS API 1.1 Producer and Consumer" to understand some baby steps to develop JMS 1.1 Producer and Consumer programs.

In this post, we are going to develop a Simple JMS 1.1 Producer and Consumer Example With Eclipse IDE and Embedded JBoss HornetQ Server.

Post Brief TOC

Introduction

As a Novice Developer to JMS API, it is bit tough to understand about JMS Servers, ConnectionFactory Creation, Queue Creation, Topic Creation etc. We will discuss these concepts in detail in coming posts. First of all, we should understand how to write a simple JMS Producer and Consumer programs without much effort.

To develop this example, we are going to use Eclipse IDE, JMS 1.1 API, Maven and JBoss Embedded HornetQ JMS Server. It is very easy to configure ConnectionFactory, Queue, Topic etc. with this embedded server. We just need to configure some XML files.

To use Embedded JMS Server, we don't need to download and install any software. We just need to configure some Jars in our Maven Project. Let's start developing application now.

Steps to Develop a JMS V.1.1 Example:

Create a Mavenized Java project in Eclipse IDE

Project Name: **EmbeddedJMS1.1Example**

Use the following pom.xml file.

```
<project xmlns="https://maven.apache.org/POM/4.0.0"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.journaldev.jms</groupId>
  <artifactId>EmbeddedJMS1.1Example</artifactId>
  <version>1.0.0</version>

  <dependencies>
        <dependency>
                <groupId>org.hornetq</groupId>
                <artifactId>hornetq-core</artifactId>
                <version>2.3.0.BETA1</version>
        </dependency>
        <dependency>
                <groupId>org.hornetq</groupId>
                <artifactId>hornetq-jms-client</artifactId>
                <version>2.3.0.BETA1</version>
        </dependency>
                <dependency>
                <groupId>org.hornetq</groupId>
                <artifactId>hornetq-jms-server</artifactId>
                <version>2.3.0.BETA1</version>
        </dependency>
        <dependency>
                <groupId>javax.jms</groupId>
                <artifactId>javax.jms-api</artifactId>
                <version>1.1</version>
```

NOTE:-This pom.xml contains 3 dependencies

- hornetq-core-x.x.jar:- It contains to provide base API for both JMS server and client programs.
- hornetq-jms-server-x.x.jar:- It contains API to provide Embedded JBoss HornetQ Server.
- hornetq-core-x.x.jar:- It contains JMS API implementation for JMS Session,Queue,Topic etc.

Configure one user in "hornetq-users.xml"t o run this program.

hornetq-users.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="urn:hornetq"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hornetq /schema/hornetq-users.xsd">
    <defaultuser name="jduser" password="jduser">
        <role name="jduser"/>
    </defaultuser>
</configuration>
```

Here we have defined one user "jduser" to use them in HornetQ Configuration XML file to configure Security.

Configure in-memory server configuration "hornetq-configuration.xml" to run this program. It is used to configure In-Memory JMS Server.

hornetq-configuration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="urn:hornetq" xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hornetq /schema/hornetq-configuration.xsd">

    <persistence-enabled>false</persistence-enabled>
    <connectors>
        <connector name="in-vm">
            <factory-class>org.hornetq.core.remoting.impl.invm.InVMConnectorFactory</factory-class>
        </connector>
    </connectors>
    <acceptors>
        <acceptor name="in-vm">
            <factory-class>org.hornetq.core.remoting.impl.invm.InVMAcceptorFactory</factory-class>
        </acceptor>
    </acceptors>
    <security-settings>
        <security-setting match="#">
            <permission type="consume" roles="jduser"/>
            <permission type="send" roles="jduser"/>
        </security-setting>
    </security-settings>

</config
```

Here We have configured JBoss HornetQ In-Memory Server and Security details.

Configure in-memory server ConnectionFactory and Queue "hornetq-jms.xml" to run this program. It is used to configure JMS Settings.

hornetq-jms.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="urn:hornetq"
xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hornetq /schema/hornetq-jms.xsd">
    <connection-factory name="JDConnectionFactory">
        <connectors>
            <connector-ref connector-name="in-vm"/>
        </connectors>
        <entries>
            <entry name="JDConnectionFactory"/>
        </entries>
    </connection-factory>

    <queue name="JDQueue">
        <entry name="/queue/JDQueue"/>
    </queue>
</configuration>
```

Here we have configured ConnectionFactory with "JDConnectionFactory" name and Queue destination with "/queue/JDQueue" JNDI name. We have linked these two things to previous configured JBoss HornetQ In-Memory Server.

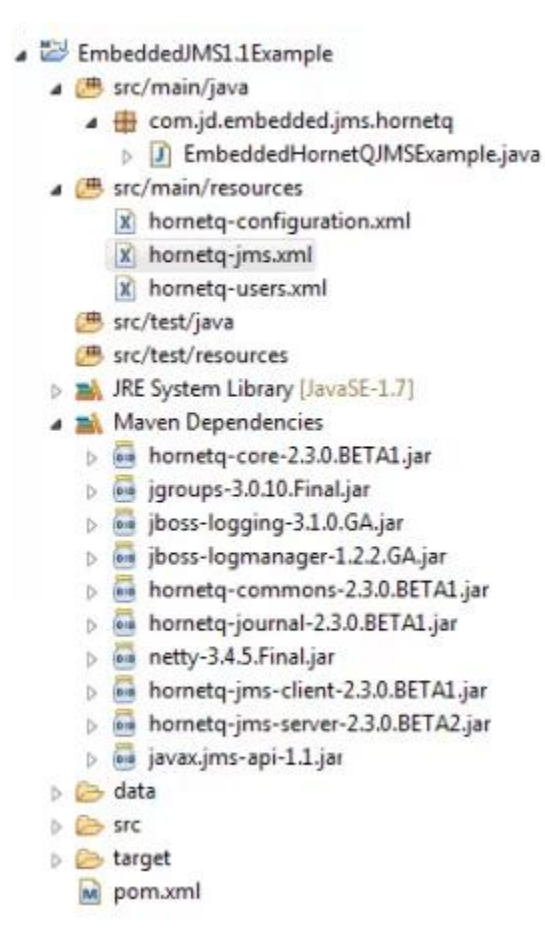
Create a Simple Java class **"EmbeddedHornetQJMSExample"** and copy below sample code.

```
package com.jd.embedded.jms.hornetq;

import javax.jms.*;
import org.hornetq.jms.server.embedded.EmbeddedJMS;

public class EmbeddedHornetQJMSExample
{
   public static void main(final String[] args)
   {
      try
      {
         // This EmbeddedJMS class acts as a JMSServer for example: JBoss HornetQ Server
         EmbeddedJMS jmsServer = new EmbeddedJMS();
         jmsServer.start();
         System.out.println("JD: Embedded JMS Server started.");
         ConnectionFactory connectionFactory =
                    (ConnectionFactory)jmsServer.lookup("JDConnectionFactory");
         Queue queue = (Queue)jmsServer.lookup("/queue/JDQueue");
         Connection connection = null;
         try
         {
            connection = connectionFactory.createConnection();
            Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);

            //1. Producer Code Start
            MessageProducer producer = session.createProducer(queue);
            TextMessage message = session.createTextMessage("Hello, I'm from JournalDev(JD).");
            System.out.println("Sending message: " + message.getText());
            producer.send(message);
```

Just for simplicity, I've created both Message Producer and Consumer in a single Java file. Please go through those two sub-sections(separated by comments) for understanding purpose.

Final Project Structure:-

If we observe the project in Eclipse IDE, it's final structure looks like as below:



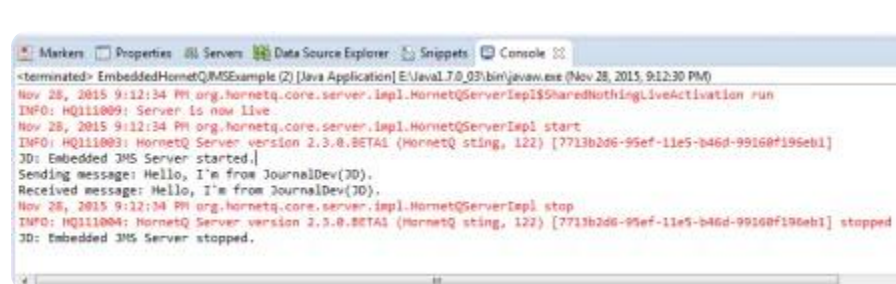
Execute and Observe the Output:-

In this section, we will run our JMS Program and observe the results in Eclipse IDE Console.

Right click on "EmbeddedHornetQJMSExample.java" Program

Run it as "Java Application"

Observe the output in Eclipse IDE Console



Here we can see both Producer Sent message and Consumer Received message.

That's it all about developing Simple JMS API 1.1 Producer and Consumer application with Embedded JBoss HornetQ Server. We will discuss JMS API 2.0 Producer and Consumer Example in my coming posts.

Please drop me a comment if you like my post or have any issues/suggestions.

Facebook Twitter WhatsApp Reddit LinkedIn Email

JMS API 2.0 Producer and Consumer
PREV

JMS 1.1 Producer and Consumer Example With Eclipse IDE, EJB Project and JBoss 6.0 AS
NEXT

Comments

Yogesh Dhavan says:
December 11, 2019 at 12:44 am

Want to do the same with Topic . Tried with same code.
COnmfigured Topic in hornetq-jms.xml file.

Reply

Romain says:
August 14, 2018 at 10:02 am

Hello,

thank you for this tutorial. If I may add my 2 cents:
– The closing tag of hornetq-configuration.xml is incomplete
– The POM's imports are now:

org.hornetq
hornetq-core
2.3.0.BETA1

org.hornetq
hornetq-jms-client
2.3.0.BETA1

org.hornetq
hornetq-jms-server
2.3.0.BETA2

javax.jms
jms
1.1

Thank you again for your help in my understanding of HornetQ

Reply

Leave a Reply