Лабораторная работа №1

Знакомство с технологией JMS (Java Messaging Service)

Цель работы

Ознакомиться с технологией JMS. Установить и ознакомиться с сервером Apache ActiveMQ.

Общие сведения

В данной лабораторной работе, а также последующих лабораторных работах, будет рассматриваться технология Java Message Service (JMS). JMS – это система передачи сообщений, разработанная компанией Sun, позволяющая разработчикам создавать гибкие и слабосвязанные приложения с использованием асинхронного обмена данными между приложениями (клиентами/серверами) через посредника. Асинхронность – это главная причина создания и использования JMS.

Общий принцип работы с JMS весьма простой: некое приложение (поставщик) хочет передать данные другому приложению, или даже нескольким приложениям (получатель). Поставщик отправляет свое сообщение на JMS сервер, получатель же проверяет JMS сервер на наличие для него сообщений и считывает их, если таковые имеются.

В жестком связывании обмен происходил бы без использования посредника напрямую, а поставщик устанавливал соединение с получателем и передавал ему сообщение, используя данное соединение.

При жестком связывании имеются две главные проблемы, которые решает JMS:

1) «один поставщик – один получатель» – для реализации многопользовательской рассылки, поставщику необходимо установить соединение с каждым из получателей;

2) «синхронность» – передача сообщения поставщиком возможна только при рабочем получателе и наличия с ним соединения.

При использовании JMS вовсе не обязательно условие рабочего получателя. Поставщик отправляет сообщение на сервер, где оно хранится указанное время или до его получения получателем. JMS сервер может самостоятельно рассылать одно сообщение всем получателям (если получателей больше 1), что снимает большой объем работы с поставщика.

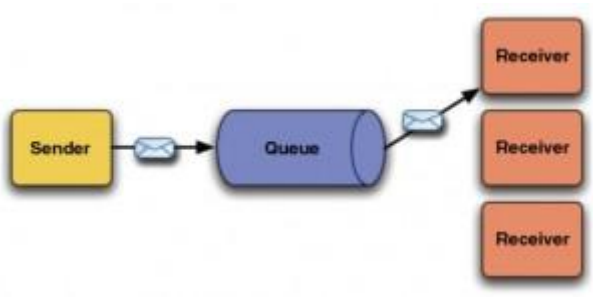
Благодаря описанным выше преимуществам, возможна асинхронная работа нескольких связанных приложений с параллельной обработкой информации.

В JMS имеется две модели передачи сообщений и соответственно для этого используются два вида объекта:

1) модель «Точка-Точка (Point-to-Point)» и объект Queue;

2) модель «Подписчик-Издатель (Publisher-Subscriber)» и объект Topic.

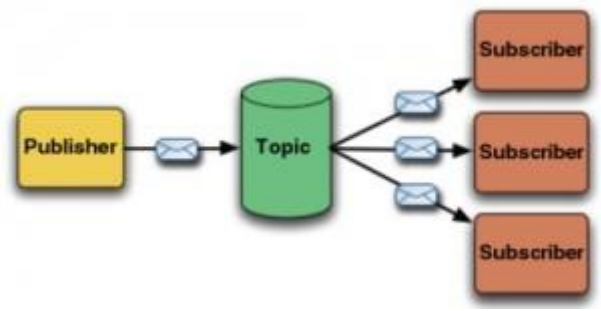
В модели «Точка-Точка» одно сообщение имеет одного и только одного получателя (классика корпоративного обмена сообщениями). Если к объекту Queue подключено больше одного получателя, ожидающего сообщения, то сообщение дойдет только к одному из получателей – тому, кто раньше его запросит (если не указан конкретный получать).



Каждое сообщение в очереди может быть считано только одним получателем, при этом оно выталкивается из очереди. Сообщение может быть прочитано когда угодно. Если получатель не работал в момент

отсылки, сообщение не пропадёт. После получения сообщения адресат посылает извещение. Таким образом, очередь является «каналом» передачи сообщения единственному получателю.

В модели «Подписчик-Издатель» сообщение получает каждый подписчик.



Подписчик подписывается на определённую «тему» Издатель публикует своё сообщение. Брокер размножает его и рассылает всем подписчикам этой темы. Получатель должен работать и быть подписан в момент отправки сообщения. То есть на тему может подписаться произвольное количество получателей и каждый из них получит экземпляр сообщения.

Адресаты могут получать свои сообщения в двух режимах – синхронном и асинхронном. Это справедливо как для очередей, так и для тем.

В синхронном режиме код программы получателя явно вызывает метод получения сообщения (receive). Если сообщение в соответствующем целевом объекте (queue или topic) отсутствует, вызывающий поток будет блокирован до его появления. В спецификации существуют перегрузки метода receive позволяющие задать период ожидания.

В асинхронном режиме разработчик реализует в классе получателя интерфейс javax.jms.MessageListener. Этот интерфейс содержит всего один

метод onMessage(), который будет вызван при появлении в целевом объекте сообщения.

Важнейшим понятием JMS является сессия (session). Можно сказать, что сессия это контекст потока, в котором выполняется передача сообщений. Сессии создаются в контексте, какого либо соединения (connection). В свою очередь, сессия является фабрикой для объектов-отправителей сообщений (producer), объектов-получателей сообщений (consumer) и самих сообщений. При передаче сообщения выполняется его сериализация. Сессии бывают транзакционные и нетранзакционные.

В JMS возможны транзакционные отправки сообщения. В этом случае сообщения группируются и отправляются внутри одной транзакции. Если произошла ошибка, то все действия отменяются, вся отправка и получение сообщений, которые были внутри транзакций, аннулируются. Чтобы воспользоваться транзакцией, необходимо создать объект Session, поддерживающий транзакцию.

Процесс выполнения работы

Основным «столпом» и центральной частью технологии является сервер сообщений, обычно именуемый брокером (message broker). Используемые в Java EE проектах серверы Glassfish и JBoss поддерживают возможность реализации JMS. Однако в реальных условиях разработки, поставляемый вместе с серверами стандартный функционал JMS практически не используется. Как правило, сервер приложений и сервер сообщений – это два абсолютно разных процесса, расположенных на различных машинах.

Зачастую разработчику приходится сталкиваться с другими популярными MQ (Messages Queue) серверами, такими как:

1) Apache ActiveMQ Server;

2) WebShpere MQ Server.

Для разрабатываемого приложения не имеет значения, какому из серверов отправлять сообщения, внутренний код программы не изменится, или изменится незначительно, так как JMS – это спецификация, и методы работы с JMS меняться не должны.

В данных лабораторных работах будет рассмотрено использование сервера Apache ActiveMQ.

Для установки и начала работы с сервером ActiveMQ необходимо выполнить следующие шаги:

1) загрузить архив с сервером, который расположен по следующей ссылке http://activemq.apache.org/activemq-5141-release.html (рисунок 1);

ActiveMQ 5.14.1 Release

Apache ActiveMQ 5.14.1 includes many resolved issues and bug fixes.

Getting the Binary Distributions

Description	Download Link	*Verify*
Windows Distribution	apache-activemq-5.14.1-bin.zip	ASC, MD5, SHA1
Unix/Linux/Cygwin Distribution	apache-activemq-5.14.1-bin.tar.gz	ASC, MD5, SHA1

Рисунок 1

2) загруженный архив необходимо разархивировать, перейти в папку bin (рисунок 2) и запустить файл activemq.bat (рисунок 3);

bin	15.09.2017 22:59	Папка с файлами	
conf	15.09.2017 23:00	Папка с файлами	
data	15.09.2017 23:00	Папка с файлами	
docs	15.09.2017 23:00	Папка с файлами	
examples	15.09.2017 23:00	Папка с файлами	
lib	15.09.2017 23:00	Папка с файлами	
webapps	15.09.2017 23:00	Папка с файлами	
webapps-demo	27.09.2016 13:14	Папка с файлами	
activemq-all-5.14.1.jar	15.09.2017 22:59	Executable Jar File	15 821 КБ
LICENSE	15.09.2017 22:59	Файл	41 КБ

Рисунок 2

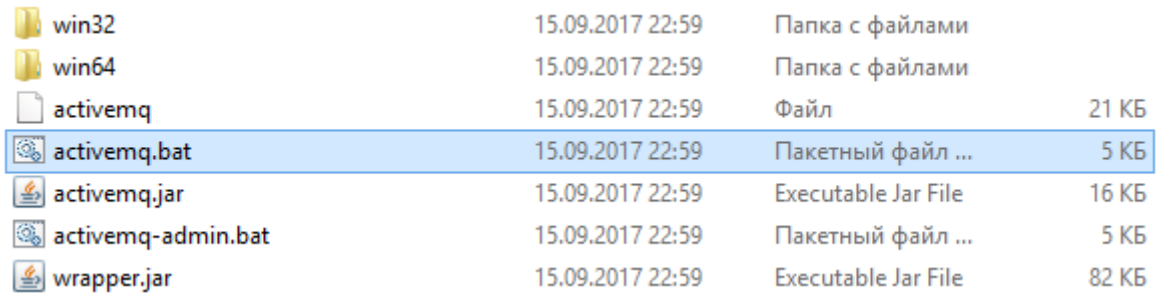


Рисунок 3

3) после этого сервер ApacheMQ будет запущен и готов принимать сообщения (рисунок 4);

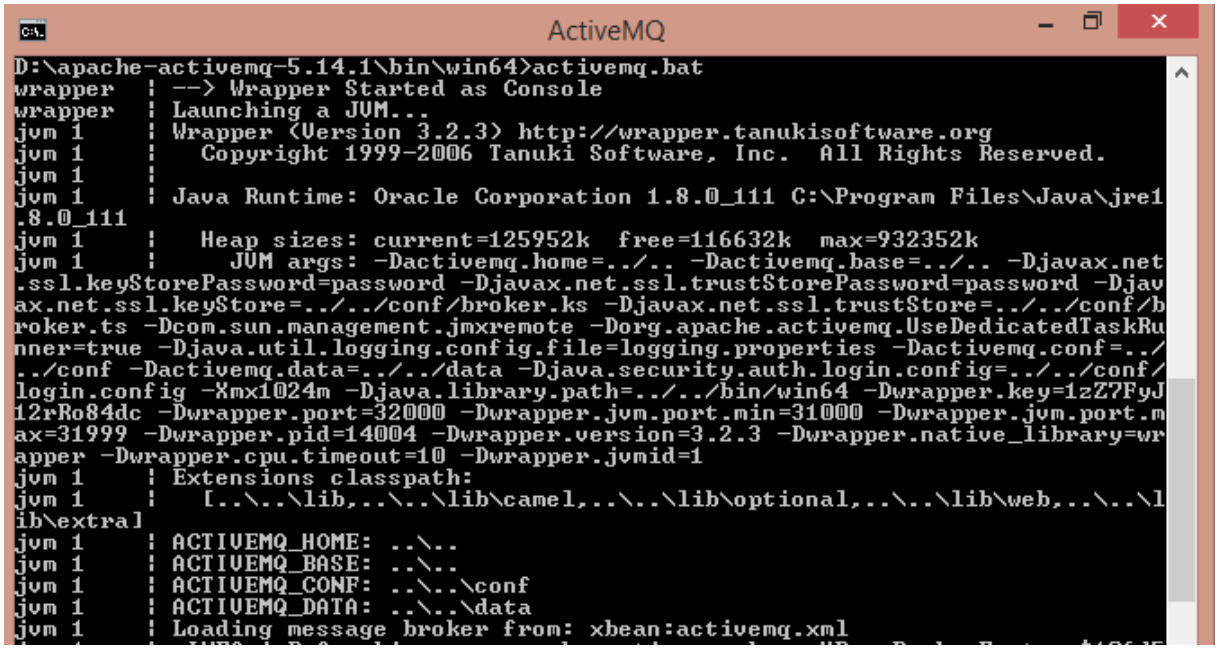


Рисунок 4

4) после запуска сервера, пользователю будет доступна веб-консоль, предназначенная для управления Active MQ, которая доступна по адресу http://localhost:8161 (рисунок 5);

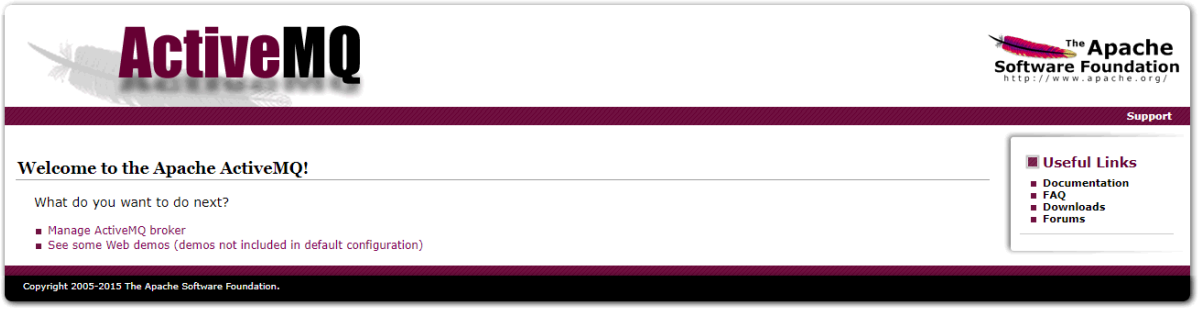


Рисунок 5

5) для ознакомления с возможностями отправки и получения сообщений, необходимо перейти по ссылке «Manage ActiveMQ broker», используя «admin/admin» в качестве логина и пароля соответственно (рисунок 6);

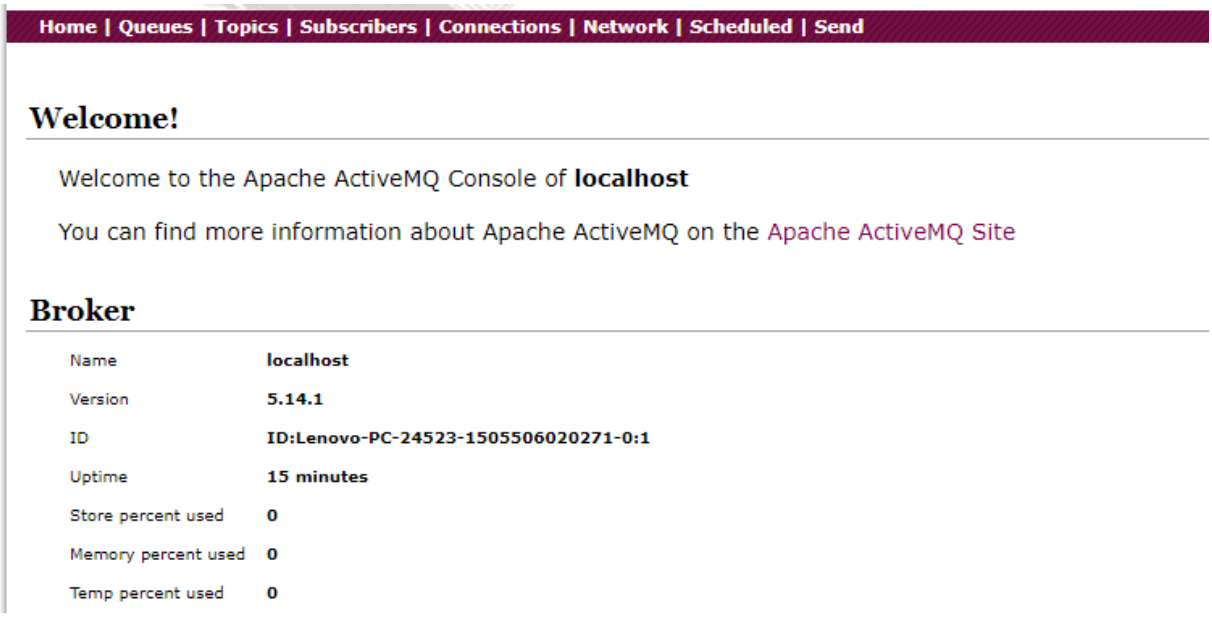


Рисунок 6

6) после чего, перейти по ссылке «Send», ввести текст «Test message» в поле «Message body» и нажать кнопку «Send» (рисунок 7);

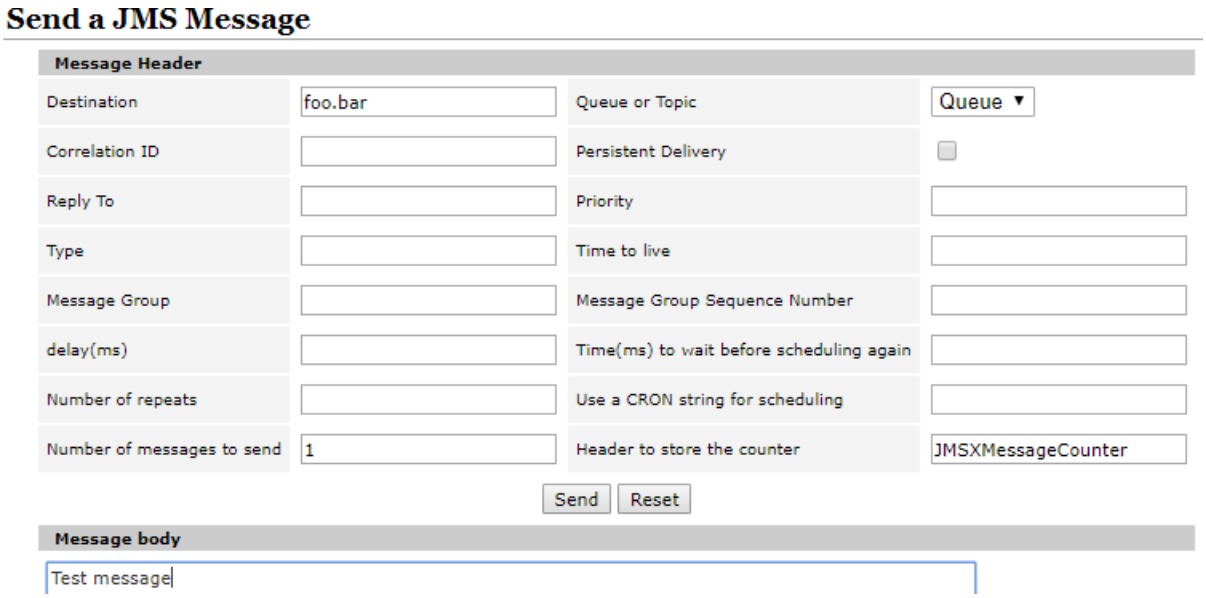


Рисунок 7

7) затем нажать на кнопку «atom» и просмотреть отправленное сообщение (рисунок 8).

Рисунок 8

В отчете необходимо кратко описать основные этапы выполнения лабораторной работы, дать ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое JMS?
2. В чем заключается общий принцип работы с JMS?
3. Какие проблемы решает JMS?

4. Какие модели передачи сообщений существуют в JMS? В чем их различие?

6. Что такое сессия?

5. Что такое транзакционная отправка сообщений в JMS?

Дополнительные источники информации

1. Java Messaging Service (JMS). http://www.oracle.com/technetwork/java/jms/index.html

2. Curry, E. (2004). Message-Oriented Middleware. In Middleware for Communications, ed. Qusay H Mahmoud. Chichester, England: John Wiley and Sons, pp. 1–28.

3. Richards, M., Richard M. H., David A. C. (2009). Java Message Service, Second Edition. O'Reilly.

4. Apache ActiveMQ. http://activemq.apache.org/

5. Snyder, B., Bosanac, D., Davies, R. (2010). ActiveMQ in Action (1st ed.). Manning Publications, p. 375.