一 .安装运行ActiveMQ：

1.下载activemq

wget <http://archive.apache.org/dist/activemq/apache-activemq/5.9.0/apache-activemq-5.9.0-bin.tar.gz>



2.解压

tar -xf apache-activemq-5.9.0-bin.tar.gz

IMG_257

[zcw@g1 ~]$ cd apache-activemq-5.9.0

[zcw@g1 apache-activemq-5.9.0]$ cd bin/

3.运行：

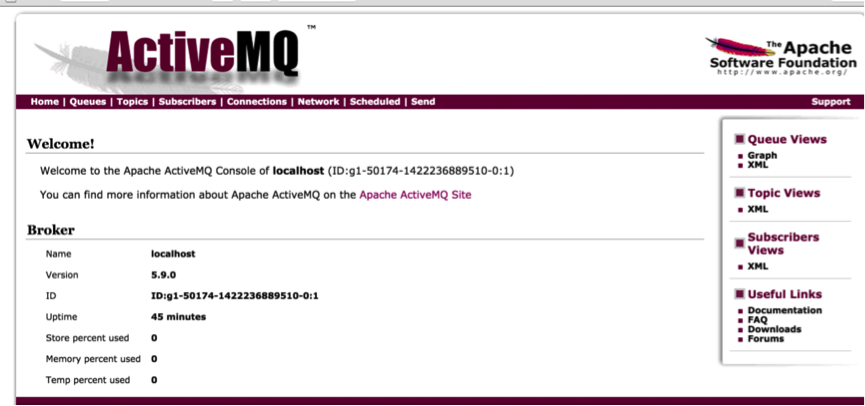
[zcw@g1 bin]$ activemq start



三种运行方式：  
(1)普通启动 ./activemq start  
(2)启动并指定日志文件 ./activemq start &gt;tmp/smlog  
(3)后台启动方式nohup ./activemq start &gt;/tmp/smlog  
前两种方式下在命令行窗口关闭时或者ctrl+c时导致进程退出，采用后台启动方式则可以避免这种情况

管理后台为：

http://ip:8161/admin/

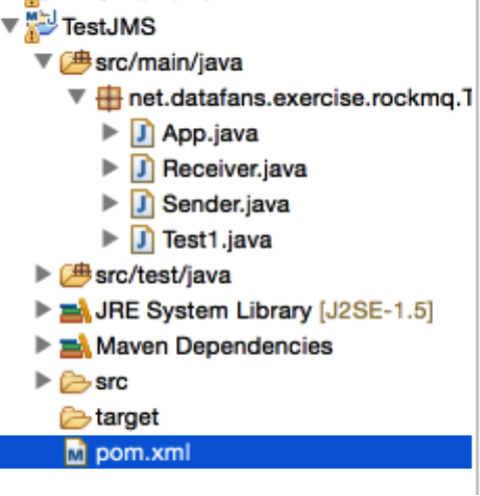


4.检查已经启动  
 ActiveMQ默认采用61616端口提供JMS服务，使用8161端口提供管理控制台服务，执行以下命令以便检验是否已经成功启动ActiveMQ服务。  
打开端口：nc -lp 61616 &  
查看哪些端口被打开 netstat -anp  
查看61616端口是否打开： netstat -an | grep 61616  
检查是否已经启动：  
(1).查看控制台输出或者日志文件   
(2).直接访问activemq的管理页面：<http://localhost:8161/admin/>

5.关闭  
如果开启方式是使用(1)或(2)，则直接ctrl+c或者关闭对应的终端即可   
如果开启方式是(3),则稍微麻烦一点：   
先查找到activemq对应的进程：   
ps -ef | grep activemq   
然后把对应的进程杀掉，假设找到的进程编号为 168168   
kill 168168

**二.创建ActiveMQ的Eclipse项目并运行**

1）P2P方式



到中心仓库（

http://search.maven.org/

）里面找到：activemq-core

pom.xml:

IMG_261

[IMG_262](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>net.datafans.exercise.rockmq</groupId>

<artifactId>TestJMS</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>TestJMS</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.apache.activemq</groupId>

<artifactId>activemq-core</artifactId>

<version>5.7.0</version>

</dependency>

</dependencies>

</project>

[IMG_263](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

Sender：

[IMG_264](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import javax.jms.Connection;

import javax.jms.ConnectionFactory;

import javax.jms.DeliveryMode;

import javax.jms.Destination;

import javax.jms.MessageProducer;

import javax.jms.Session;

import javax.jms.TextMessage;

import org.apache.activemq.ActiveMQConnection;

import org.apache.activemq.ActiveMQConnectionFactory;

public class Sender {

private static final int SEND\_NUMBER = 5;

public static void main(String[] args) {

// ConnectionFactory ：连接工厂，JMS 用它创建连接 ConnectionFactory connectionFactory;

// Connection ：JMS 客户端到JMS Provider 的连接

Connection connection = null;

// Session： 一个发送或接收消息的线程 Session session;

// Destination ：消息的目的地;消息发送给谁. Destination destination;

// MessageProducer：消息发送者 MessageProducer producer;

// TextMessage message;

// 构造ConnectionFactory实例对象，此处采用ActiveMq的实现jar

connectionFactory = new ActiveMQConnectionFactory(

ActiveMQConnection.DEFAULT\_USER,

ActiveMQConnection.DEFAULT\_PASSWORD,

"tcp://ip:61616");

try {

// 构造从工厂得到连接对象

connection = connectionFactory.createConnection();

// 启动 connection.start();

// 获取操作连接

session = connection.createSession(Boolean.TRUE,

Session.AUTO\_ACKNOWLEDGE);

// 获取session注意参数值xingbo.xu-queue是一个服务器的queue，须在在ActiveMq的console配置

destination = session.createQueue("FirstQueue");

// 得到消息生成者【发送者】

producer = session.createProducer(destination);

// 设置不持久化，此处学习，实际根据项目决定 producer.setDeliveryMode(DeliveryMode.NON\_PERSISTENT);

// 构造消息，此处写死，项目就是参数，或者方法获取 sendMessage(session, producer);

session.commit();

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (null != connection)

connection.close();

} catch (Throwable ignore) {

}

}

}

public static void sendMessage(Session session, MessageProducer producer)

throws Exception {

for (int i = 1; i <= SEND\_NUMBER; i++) {

TextMessage message = session

.createTextMessage("ActiveMq 发送的消息" + i);

// 发送消息到目的地方

System.out.println("发送消息：" + "ActiveMq 发送的消息" + i);

producer.send(message);

}

}

}

[IMG_265](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

Receiver：

[IMG_266](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import javax.jms.Connection;

import javax.jms.ConnectionFactory;

import javax.jms.Destination;

import javax.jms.MessageConsumer;

import javax.jms.Session;

import javax.jms.TextMessage;

import org.apache.activemq.ActiveMQConnection;

import org.apache.activemq.ActiveMQConnectionFactory;

public class Receiver {

public static void main(String[] args) {

// ConnectionFactory ：连接工厂，JMS 用它创建连接 ConnectionFactory connectionFactory;

// Connection ：JMS 客户端到JMS Provider 的连接

Connection connection = null;

// Session： 一个发送或接收消息的线程 Session session;

// Destination ：消息的目的地;消息发送给谁. Destination destination;

// 消费者，消息接收者 MessageConsumer consumer;

connectionFactory = new ActiveMQConnectionFactory(

ActiveMQConnection.DEFAULT\_USER,

ActiveMQConnection.DEFAULT\_PASSWORD,

"tcp://ip:61616");

try {

// 构造从工厂得到连接对象

connection = connectionFactory.createConnection();

// 启动 connection.start();

// 获取操作连接

session = connection.createSession(Boolean.FALSE,

Session.AUTO\_ACKNOWLEDGE);

// 获取session注意参数值xingbo.xu-queue是一个服务器的queue，须在在ActiveMq的console配置

destination = session.createQueue("FirstQueue");

consumer = session.createConsumer(destination);

while (true) {

//设置接收者接收消息的时间，为了便于测试，这里谁定为100s

TextMessage message = (TextMessage) consumer.receive(100000);

if (null != message) {

System.out.println("收到消息" + message.getText());

} else {

break;

}

}

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (null != connection)

connection.close();

} catch (Throwable ignore) {

}

}

}

}

[IMG_267](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

测试过程：

发送



接收



 2）Pub/Sub模式

[IMG_270](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import javax.jms.Connection;

import javax.jms.ConnectionFactory;

import javax.jms.DeliveryMode;

import javax.jms.Destination;

import javax.jms.MessageProducer;

import javax.jms.Session;

import javax.jms.TextMessage;

import javax.jms.Topic;

import org.apache.activemq.ActiveMQConnection;

import org.apache.activemq.ActiveMQConnectionFactory;

public class Pub {

private static final int SEND\_NUMBER = 5;

public static void main(String[] args) {

// ConnectionFactory ：连接工厂，JMS 用它创建连接 ConnectionFactory connectionFactory;

// Connection ：JMS 客户端到JMS Provider 的连接

Connection connection = null;

// Session： 一个发送或接收消息的线程 Session session;

// Destination ：消息的目的地;消息发送给谁. Destination destination;

// MessageProducer：消息发送者 MessageProducer producer;

// TextMessage message;

// 构造ConnectionFactory实例对象，此处采用ActiveMq的实现jar

connectionFactory = new ActiveMQConnectionFactory(

ActiveMQConnection.DEFAULT\_USER,

ActiveMQConnection.DEFAULT\_PASSWORD,

"tcp://ip:61616");

try {

// 构造从工厂得到连接对象

connection = connectionFactory.createConnection();

// 启动 connection.start();

// 获取操作连接

session = connection.createSession(Boolean.TRUE,

Session.AUTO\_ACKNOWLEDGE);

Topic topic = session.createTopic("MessageTopic");

producer = session.createProducer(topic);

producer.setDeliveryMode(DeliveryMode.NON\_PERSISTENT);

TextMessage message = session.createTextMessage();

message.setText("message\_hello\_chenkangxian");

producer.send(message);

// // 获取session注意参数值xingbo.xu-queue是一个服务器的queue，须在在ActiveMq的console配置// destination = session.createQueue("FirstQueue");// // 得到消息生成者【发送者】// producer = session.createProducer(destination);// // 设置不持久化，此处学习，实际根据项目决定// producer.setDeliveryMode(DeliveryMode.NON\_PERSISTENT);// // 构造消息，此处写死，项目就是参数，或者方法获取// sendMessage(session, producer);// session.commit();

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (null != connection)

connection.close();

} catch (Throwable ignore) {

}

}

}

public static void sendMessage(Session session, MessageProducer producer)

throws Exception {

for (int i = 1; i <= SEND\_NUMBER; i++) {

TextMessage message = session

.createTextMessage("ActiveMq 发送的消息" + i);

// 发送消息到目的地方

System.out.println("发送消息：" + "ActiveMq 发送的消息" + i);

producer.send(message);

}

}

}

[IMG_271](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

[IMG_272](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import javax.jms.Connection;

import javax.jms.ConnectionFactory;

import javax.jms.Destination;

import javax.jms.JMSException;

import javax.jms.Message;

import javax.jms.MessageConsumer;

import javax.jms.MessageListener;

import javax.jms.Session;

import javax.jms.TextMessage;

import javax.jms.Topic;

import org.apache.activemq.ActiveMQConnection;

import org.apache.activemq.ActiveMQConnectionFactory;

public class Sub {

public static void main(String[] args) {

// ConnectionFactory ：连接工厂，JMS 用它创建连接 ConnectionFactory connectionFactory;

// Connection ：JMS 客户端到JMS Provider 的连接

Connection connection = null;

// Session： 一个发送或接收消息的线程 Session session;

// Destination ：消息的目的地;消息发送给谁. Destination destination;

// 消费者，消息接收者 MessageConsumer consumer;

connectionFactory = new ActiveMQConnectionFactory(

ActiveMQConnection.DEFAULT\_USER,

ActiveMQConnection.DEFAULT\_PASSWORD,

"tcp://ip:61616");

try {

// 构造从工厂得到连接对象

connection = connectionFactory.createConnection();

// 启动 connection.start();

// 获取操作连接

session = connection.createSession(Boolean.TRUE,

Session.AUTO\_ACKNOWLEDGE);

Topic topic = session.createTopic("MessageTopic");

consumer = session.createConsumer(topic);

consumer.setMessageListener(new MessageListener() {

public void onMessage(Message message) {

// TODO Auto-generated method stub

TextMessage tm = (TextMessage)message;

try {

System.out.println(tm.getText());

} catch (JMSException e) {

// TODO Auto-generated catch block e.printStackTrace();

}

}

});

// 获取session注意参数值xingbo.xu-queue是一个服务器的queue，须在在ActiveMq的console配置// destination = session.createQueue("FirstQueue");// consumer = session.createConsumer(destination);// while (true) {// //设置接收者接收消息的时间，为了便于测试，这里谁定为100s// TextMessage message = (TextMessage) consumer.receive(100000);// if (null != message) {// System.out.println("收到消息" + message.getText());// } else {// break;// }// }

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (null != connection)

connection.close();

} catch (Throwable ignore) {

}

}

}

}

[IMG_273](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

测试

PS：代码都写出来了只是不知道为啥测试Pub/Sub模式一直都出不来

3.Request和Reply模式

CONFIG：

[IMG_274](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

/\*\*

\* @author mike

\* @date Apr 17, 2014

\*/

package net.datafans.exercise.rockmq.TestJMS;

class CONFIG {

public static final java.lang.String AUTHOR = "Mike Tang";

public static final java.lang.String QUEUE\_NAME = "REQUEST AND REPLY";

public static void introduce() {

java.lang.StringBuilder builder = new java.lang.StringBuilder();

builder.append("This is a simple example to show how to write a \n");

builder.append("request and reply pattern program with Apache ActiveMQ.\n");

builder.append("which is not support such pattern.\n\n");

builder.append(" -------- By Mike Tang\n");

builder.append(" 2014-4-17 at Soochow University\n");

System.out.println(builder.toString());

}

public static void introduceServer() {

System.out.println("Wait for the client send messages, and you will see something");

}

public static void introduceClient() {

System.out.println("Input something and ENTER, you will get the reply.\n"

+ "and input 'exit' stop the program.");

}

}

[IMG_275](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

MessageClient

[IMG_276](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import java.util.Scanner;

import java.util.UUID;

import javax.jms.Connection;

import javax.jms.DeliveryMode;

import javax.jms.Destination;

import javax.jms.JMSException;

import javax.jms.Message;

import javax.jms.MessageConsumer;

import javax.jms.MessageListener;

import javax.jms.MessageProducer;

import javax.jms.Session;

import javax.jms.TemporaryQueue;

import javax.jms.TextMessage;

import org.apache.activemq.ActiveMQConnectionFactory;

public class MessageClient {

String brokerurl = "ip";

static Session session = null;

static MessageConsumer consumer = null;

static MessageProducer producer = null;

static TemporaryQueue temporaryQueue = null;

public void setURL(String brokerurl) {

this.brokerurl = brokerurl;

}

public void start() throws JMSException {

ActiveMQConnectionFactory connectionFactory = new ActiveMQConnectionFactory(brokerurl);

Connection connection = connectionFactory.createConnection();

connection.start();

session = connection.createSession(false, Session.AUTO\_ACKNOWLEDGE);

Destination destination = session.createQueue(CONFIG.QUEUE\_NAME);

temporaryQueue = session.createTemporaryQueue();

{

producer = session.createProducer(destination);

producer.setDeliveryMode(DeliveryMode.NON\_PERSISTENT);

consumer = session.createConsumer(temporaryQueue);

consumer.setMessageListener(new MMessageListener());

}

}

public void request(String request) throws JMSException {

System.out.println("REQUEST TO : " + request);

TextMessage textMessage = session.createTextMessage();

textMessage.setText(request);

textMessage.setJMSReplyTo(temporaryQueue);

textMessage.setJMSCorrelationID(UUID.randomUUID().toString());

MessageClient.producer.send(textMessage);

}

private static class MMessageListener implements MessageListener {

public void onMessage(Message message) {

try {

if (message instanceof TextMessage) {

String messageText = ((TextMessage) message).getText();

System.out.println("REPLY FROM : " + messageText.toUpperCase());

}

} catch (JMSException e) {

e.printStackTrace();

}

}

}

// -----------------------------------------------------------------------

public static void main(String[] args) {

CONFIG.introduce();

CONFIG.introduceClient();

try {

MessageClient client = new MessageClient();

client.setURL("tcp://ip:61616");

Scanner scanner = new Scanner(System.in);

client.start();

{

System.out.println("-----------------------------------------");

System.out.println("| Client Start! |");

System.out.println("-----------------------------------------");

}

String message = "";

int i = 0;

while (!(message = scanner.next()).equals("exit")) {

client.request(message + " : " + i++);

}

scanner.close();

} catch (JMSException e) {

e.printStackTrace();

}

}

}

[IMG_277](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

MessageServer

[IMG_278](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

package net.datafans.exercise.rockmq.TestJMS;

import javax.jms.Connection;

import javax.jms.DeliveryMode;

import javax.jms.Destination;

import javax.jms.JMSException;

import javax.jms.Message;

import javax.jms.MessageConsumer;

import javax.jms.MessageListener;

import javax.jms.MessageProducer;

import javax.jms.Session;

import javax.jms.TextMessage;

import org.apache.activemq.ActiveMQConnectionFactory;

import org.apache.activemq.broker.BrokerService;

public class MessageServer {

String brokerurl = "ip";

static BrokerService brokerService = null;

static MessageConsumer consumer = null;

static MessageProducer producer = null;

static Session session = null;

public void setURL(String brokerurl) {

this.brokerurl = brokerurl;

}

public void start() throws JMSException {

createBroker(brokerurl);

setConsumer(brokerurl);

}

public void createBroker(String brokerurl) {

try {

brokerService = new BrokerService();

brokerService.setUseJmx(false);

brokerService.setPersistent(false);

brokerService.addConnector(brokerurl);

brokerService.start();

} catch (Exception e) {

e.printStackTrace();

}

}

public void setConsumer(String url) throws JMSException {

ActiveMQConnectionFactory connectionFactory = new ActiveMQConnectionFactory(url);

Connection connection = connectionFactory.createConnection();

connection.start();

session = connection.createSession(false, Session.AUTO\_ACKNOWLEDGE);

Destination destination = session.createQueue(CONFIG.QUEUE\_NAME);

{

producer = session.createProducer(null);

producer.setDeliveryMode(DeliveryMode.NON\_PERSISTENT);

consumer = session.createConsumer(destination);

consumer.setMessageListener(new MMessageListener());

}

}

private static class MMessageListener implements MessageListener {

public void onMessage(Message message) {

try {

TextMessage response = session.createTextMessage();

if (message instanceof TextMessage) {

String messageText = ((TextMessage) message).getText();

response.setText(handleMessage(messageText));

System.out.println("REQUEST FROM " + messageText.toUpperCase());

}

response.setJMSCorrelationID(message.getJMSCorrelationID());

producer.send(message.getJMSReplyTo(), response);

} catch (JMSException e) {

e.printStackTrace();

}

}

private String handleMessage(String text) {

return "RESPONSE TO " + text.toUpperCase();

}

}

// -----------------------------------------------------------------

public static void main(String[] args) {

CONFIG.introduce();

CONFIG.introduceServer();

try {

MessageServer server = new MessageServer();

server.setURL("tcp://ip:61616");

server.start();

{

System.out.println("-----------------------------------------");

System.out.println("| Server Start! |");

System.out.println("-----------------------------------------");

}

} catch (JMSException e) {

e.printStackTrace();

}

}

}

[IMG_279](http://www.cnblogs.com/super-d2/p/javascript:void(0);)

测试：



