公告

昵称： 风一样的码农
园龄： 4年
粉丝： 815
关注： 0
+加关注

搜索

找找看

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

积分与排名

积分 - 461589

排名 - 542

随笔分类 (226)

ActiveMQ(1)

Apache Commons(6)

JMS(Java消息服务)入门教程

阅读目录

什么是Java消息服务

Java消息服务指的是两个应用程序之间进行异步通信的API，它为标准消息协议和消息服务提供了一组通用接口，包括创建、发送、读取消息等，用于支持JAVA应用程序开发。在J2EE中，当两个应用程序使用JMS进行通信时，它们之间并不是直接相连的，而是通过一个共同的消息收发服务连接起来，可以达到解耦的效果，我们将会在接下来的教程中详细介绍。

为什么需要JMS

在JAVA中，如果两个应用程序之间对各自都不了解，甚至这两个程序可能部署在不同的大洲上，那么它们之间如何发送消息呢？举个例子，一个应用程序A部署在印度，另一个应用程序部署在美国，然后每当A触发某件事后，B想从A获取一些更新信息。当然，也有可能不止一个B对A的更新信息感兴趣，可能会有N个类似B的应用程序想从A中获取更新的信息。

在这种情况下，JAVA提供了最佳的解决方案-JMS，完美解决了上面讨论的问题。

JMS同样适用于基于事件的应用程序，如聊天服务，它需要一种发布事件机制向所有与服务器连接的客户端发送消息。JMS与RMI不同，发送消息的时候，接收者不需要在线。服务器发送了消息，然后就不管了；等到客户端上线的时候，能保证接收到服务器发送的消息。这是一个很强大的解决方案，能处理当今世界很多普遍问题。

JMS的优势

异步

JMS天生就是异步的，客户端获取消息的时候，不需要主动发送请求，消息会自动发送给可用的客户端。

可靠

JMS保证消息只会递送一次。大家都遇到过重复创建消息问题，而JMS能帮你避免该问题。

JMS消息传送模型

在JMS API出现之前，大部分产品使用“点对点”和“发布/订阅”中的任一方式来进行消息通讯。JMS定义了这两种消息发送模型的规范，它们相互独立。任何JMS的提供者可以实现其中的一种或两种模型，这是它们自己的选择。JMS规范提供了通用接口保证我们基于JMS API编写的程序适用于任何一种模型。

让我们更加详细的看下这两种消息传送模型：

点对点消息传送模型

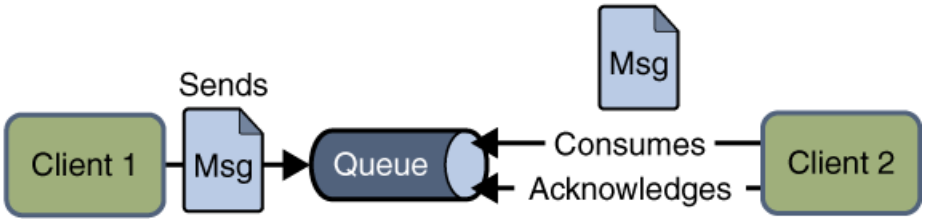
在点对点消息传送模型中，应用程序由消息队列，发送者，接收者组成。每一个消息发送给一个特殊的消息队列，该队列保存了所有发送给它的消息(除了被接收者消费掉的和过期的消息)。点对点消息模型有一些特性，如下：

每个消息只有一个接收者；

消息发送者和接收者并没有时间依赖性；

当消息发送者发送消息的时候，无论接收者程序在不在运行，都能获取到消息；

当接收者收到消息的时候，会发送确认收到通知（acknowledgement）。



发布/订阅消息传递模型

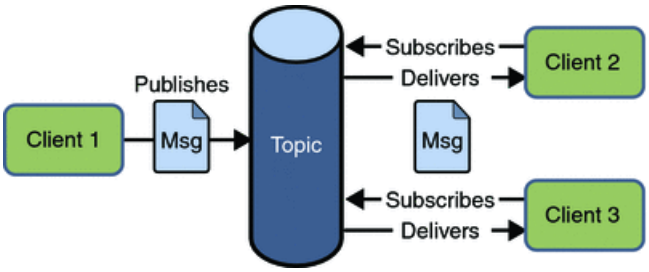
在发布/订阅消息模型中，发布者发布一个消息，该消息通过topic传递给所有的客户端。在这种模型中，发布者和订阅者彼此不知道对方，是匿名的且可以动态发布和订阅topic。topic主要用于保存和传递消息，且会一直保存消息直到消息被传递给客户端。

发布/订阅消息模型特性如下：

一个消息可以传递给多个订阅者

发布者和订阅者有时间依赖性，只有当客户端创建订阅后才能接受消息，且订阅者需一直保持活动状态以接收消息。

为了缓和这样严格的时间相关性，JMS允许订阅者创建一个可持久化的订阅。这样，即使订阅者没有被激活（运行），它也能接收到发布者的消息。



接收消息

设计模式(25)
生活随笔(2)
数据库(9)
未分类(8)
英文翻译(32)

随笔档案 (214)
2019年3月(10)
2018年10月(5)
2018年9月(5)
2018年8月(3)
2018年5月(3)
2017年8月(1)
2017年7月(2)
2017年6月(4)
2017年5月(4)
2017年3月(1)
2017年2月(5)
2017年1月(8)
2016年12月(12)
2016年11月(11)
2016年10月(9)
2016年9月(7)
2016年8月(1)
2016年7月(6)
2016年6月(13)
2016年5月(34)

在JMS中，消息的接收可以使用以下两种方式：

同步

使用同步方式接收消息的话，消息订阅者调用receive()方法。在receive()中，消息未到达或在到达指定时间之前，方法会阻塞，直到消息可用。

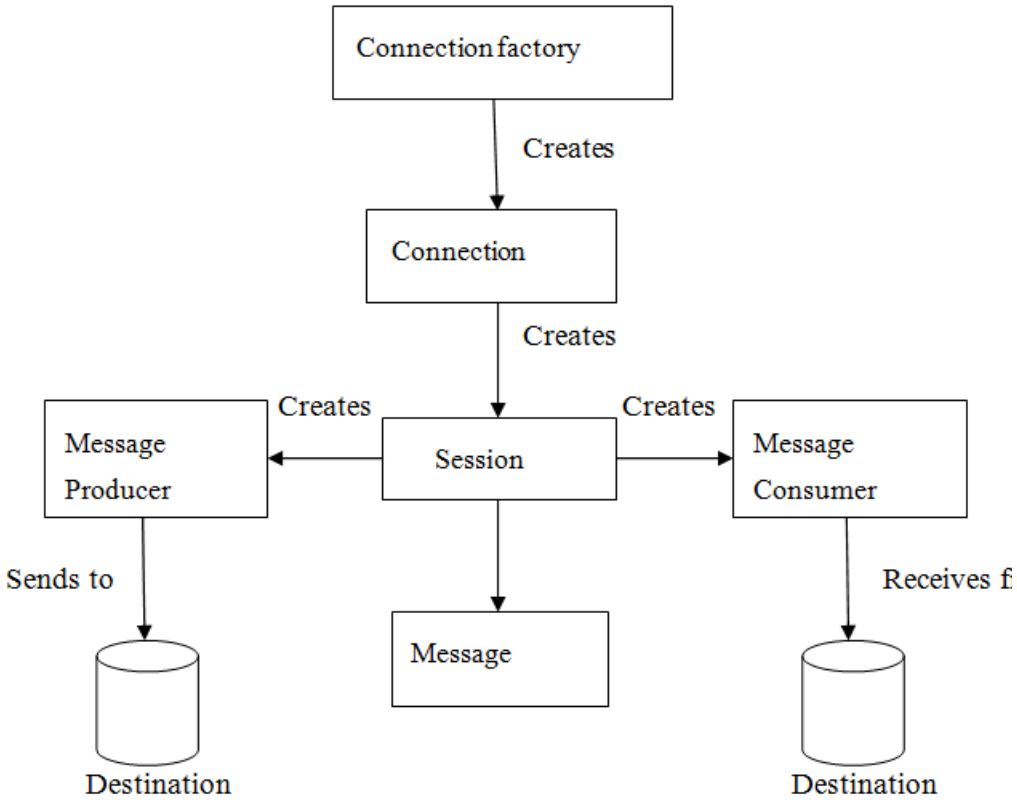
异步

使用异步方式接收消息的话，消息订阅者需注册一个消息监听者，类似于事件监听器，只要消息到达，JMS服务提供者会通过调用监听器的onMessage()递送消息。

JMS编程接口

JMS应用程序由如下基本模块组成：

1. 管理对象（Administered objects）-连接工厂（Connection Factories）和目的地（Destination）
2. 连接对象（Connections）
3. 会话（Sessions）
4. 消息生产者（Message Producers）
5. 消息消费者（Message Consumers）
6. 消息监听者（Message Listeners）



JMS管理对象

管理对象（Administered objects）是预先配置的JMS对象，由系统管理员为使用JMS的客户端创建，主要有两个被管理的对象：

2016年4月(13)

2016年3月(14)

2016年2月(21)

2016年1月(15)

2015年8月(1)

2015年6月(1)

2015年3月(1)

2014年12月(4)

阅读排行榜

1. Mysql中的存储过程(154880)

2. Java中的关键字 transient(86256)

3. 发现一个国内牛逼的maven仓库，速度真的太快了(84565)

4. Mysql中的视图(73350)

5. JMS(Java消息服务)入门教程(68820)

6. Spring Data JPA例子[基于Spring Boot、Mysql](42983)

7. Mysql 中的事件//定时任务(35694)

8. 一个简单的Java web服务器实现(35318)

9. 使用iText库创建PDF文件(32623)

10. Tomcat中的Session小结(31619)

11. jstack简单使用，定位死循环、线程阻塞、死锁等问题(25906)

12. 关于JAVA中子类和父类的构造方法(21868)

13. Spring Session - 使用Redis存储HttpSession例子(19875)

14. 【译】Spring 4 @PropertySourc

连接工厂（ConnectionFactory）

目的地（Destination）

这两个管理对象由JMS系统管理员通过使用Application Server管理控制台创建，存储在应用程序服务器的JNDI名字空间或JNDI注册表。

连接工厂（ConnectionFactory）

客户端使用一个连接工厂对象连接到JMS服务提供者，它创建了JMS服务提供者和客户端之间的连接。JMS客户端（如发送者或接受者）会在JNDI名字空间中搜索并获取该连接。使用该连接，客户端能够与目的地通讯，往队列或话题发送/接收消息。让我们用一个例子来理解如何发送消息：

```
QueueConnectionFactory queueConnFactory = (QueueConnectionFactory) initialCtx.lookup
("primaryQCF");
Queue purchaseQueue = (Queue) initialCtx.lookup ("Purchase_Queue");
Queue returnQueue = (Queue) initialCtx.lookup ("Return_Queue");
```

目的地（Destination）

目的地指明消息被发送的目的地以及客户端接收消息的来源。JMS使用两种目的地，队列和话题。如下代码指定了一个队列和话题。

创建一个队列Session

```
QueueSession ses = con.createQueueSession (false, Session.AUTO_ACKNOWLEDGE);  //get
the Queue object
Queue t = (Queue) ctx.lookup ("myQueue");  //create QueueReceiver
QueueReceiver receiver = ses.createReceiver(t);
```

创建一个话题Session

```
TopicSession ses = con.createTopicSession (false, Session.AUTO_ACKNOWLEDGE); // get
the Topic object
Topic t = (Topic) ctx.lookup ("myTopic");  //create TopicSubscriber
TopicSubscriber receiver = ses.createSubscriber(t);
```

JMS连接

连接对象封装了与JMS提供者之间的虚拟连接，如果我们有一个ConnectionFactory对象，可以使用它来创建一个连接。

```
Connection connection = connectionFactory.createConnection();
```

创建完连接后，需要在程序使用结束后关闭它：

```
connection.close();
```

JMS 会话（Session）

Session是一个单线程上下文，用于生产和消费消息，可以创建出消息生产者和消息消费者。

Session对象实现了Session接口，在创建完连接后，我们可以使用它创建Session。

```
Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
```

JMS消息生产者

消息生产者由Session创建，用于往目的地发送消息。生产者实现MessageProducer接口，我们可以为目的地、队列或话题创建生产者；

```
MessageProducer producer = session.createProducer(dest);
MessageProducer producer = session.createProducer(queue);
MessageProducer producer = session.createProducer(topic);
```

e和@Value注解示例(18403)

15. Java数组在内存中是如何存放的(17838)

评论排行榜

1. spring + spring mvc + mybatis + react + reflux + webpack Web工程例子(27)

2. Tomcat中的Session小结(15)

3. 发现一个国内牛逼的maven仓库，速度真的太快了(15)

4. HashMap源码分析(14)

5. Mysql中的视图(13)

推荐排行榜

1. Java中的关键字 transient(25)

2. Mysql中的存储过程(17)

3. Tomcat中的Session小结(14)

4. JMS(Java消息服务)入门教程(13)

5. 发现一个国内牛逼的maven仓库，速度真的太快了(12)

6. JAVA并发编程J.U.C学习总结(12)

7. Mysql中的视图(11)

8. 使用iText库创建PDF文件(9)

9. ArrayList(8)

10. JAVA 8 函数式接口 - Functional Interface(8)

11. spring + spring mvc + mybatis + react + reflux + webpack Web工程例子(7)

12. HashMap源码分析(7)

13. Spring Data JPA例子[基于Spring Boot、Mysql](6)

创建完消息生产者后，可以使用send方法发送消息：

```
producer.send(message);
```

JMS消息消费者

消息消费者由Session创建，用于接受目的地发送的消息。消费者实现MessageConsumer接口，，我们可以为目的地、队列或话题创建消费者；

```
MessageConsumer consumer = session.createConsumer(dest);
MessageConsumer consumer = session.createConsumer(queue);
MessageConsumer consumer = session.createConsumer(topic);
```

JMS消息监听器

JMS消息监听器是消息的默认事件处理者，他实现了MessageListener接口，该接口包含一个onMessage方法，在该方法中需要定义消息达到后的具体动作。通过调用setMessageListener方法我们给指定消费者定义了消息监听器

```
Listener myListener = new Listener();
consumer.setMessageListener(myListener);
```

JMS消息结构

JMS客户端使用JMS消息与系统通讯，JMS消息虽然格式简单但是非常灵活， JMS消息由三部分组成：

消息头

JMS消息头预定义了若干字段用于客户端与JMS提供者之间识别和发送消息，预编译头如下：

– JMSDestination
– JMSDeliveryMode
– JMSMessageID
– JMSTimestamp
– JMSCorrelationID
– JMSReplyTo
– JMSRedelivered
– JMSType
– JMSExpiration
– JMSPriority

消息属性

我们可以给消息设置自定义属性，这些属性主要是提供给应用程序的。对于实现消息过滤功能，消息属性非常有用，JMS API定义了一些标准属性，JMS服务提供者可以选择性的提供部分标准属性。

消息体

在消息体中，JMS API定义了五种类型的消息格式，让我们可以以不同的形式发送和接受消息，并提供了对已有消息格式的兼容。不同的消息类型如下：

Text message : javax.jms.TextMessage，表示一个文本对象。
Object message : javax.jms.ObjectMessage，表示一个JAVA对象。
Bytes message : javax.jms.BytesMessage，表示字节数据。
Stream message :javax.jms.StreamMessage，表示java原始值数据流。
Map message : javax.jms.MapMessage，表示键值对。

最后补充一下，常见的开源JMS服务的提供者，如下：

JBoss 社区所研发的 HornetQ

Joram

Coridan的MantaRay

The OpenJMS Group的OpenJMS

JMS使用示例

基于Tomcat + JNDI + ActiveMQ实现JMS的点对点消息传送

JMS发布/订阅消息传送例子

译文链接（做了部分修改~~）

http://howtodoinjava.com/jms/jms-java-message-service-tutorial/

以上就是JMS的入门教程，学习愉快~

@Author 风一样的码农
@HomePageUrl http://www.cnblogs.com/chenpi/
@Copyright 转载请注明出处，谢谢~

分类: Redis&Jms

标签: JAVA, jms, ActiveMQ, JNDI

好文要顶 关注我 收藏该文

风一样的码农
关注 - 0
粉丝 - 815
+加关注

13 0

« 上一篇： 二叉树的先序、中序、后序遍历
» 下一篇： 基于Tomcat + JNDI + ActiveMQ实现JMS的点对点消息传送

posted @ 2016-06-06 17:23 风一样的码农 阅读(68820) 评论(9) 编辑 收藏

评论列表

#1楼 2017-02-24 14:17 我不是外星人

看起来好复杂，还是不太理解

支持(0) 反对(2)

分析的很透彻，不错。

支持(0) 反对(2)

#3楼 2017-10-20 23:26 final变量

JMS(Java消息服务)入门教程- 风一样的码农- 博客园，写的不错不错，收藏了。

推荐下，分库分表中间件 Sharding-JDBC 源码解析 17 篇：http://www.yunai.me/categories/Sharding-JDBC/?cnblog&602

淞

支持(0) 反对(4)

#4楼 2018-03-22 17:12 一根指

可以，学习了！

支持(0) 反对(0)

#5楼 2018-06-18 11:02 Mr凯撒

写的不错，老哥！

支持(0) 反对(0)

#6楼 2019-01-31 16:11 anll

很棒~~~

支持(0) 反对(0)

#7楼 2019-05-05 11:37 iisme

博客园样式不错啊~

支持(0) 反对(0)

#8楼 2019-10-11 22:31 拉桑

> 引用
> 当消息发送者发送消息的时候，无论接收者程序在不在运行，都能获取到消息；

应该是
“当消息发送者发送消息的时候，无论接收者程序在不在运行，发送者都能发送消息；”吧

支持(0) 反对(0)

#9楼 2019-12-06 11:27 jupiter_gao

很全面， 很好，谢谢你

支持(0) 反对(0)

刷新评论 刷新页面 返回

注册用户登录后才能发表评论，请 登录 或 注册， 访问 网站首页。

Copyright © 2020 风一样的码农
Powered by .NET Core 3.1.1 on Linux