

廈門大學



软件学院

《中间件技术》实验报告

题 目 中间件技术实验一

姓 名 陈澄

学 号 32420212202930

班 级 软工三班

实验时间 2024/2/29

2024 年 2 月 29 日

1 实验目的

- 1) 理解中间件的基本概念
- 2) 理解消息型中间件的基本概念
- 3) 掌握基于中间件技术进行简单开发的基本过程。

2 实验环境

操作系统: Windows11

编译环境: IntelliJ IDEA

平台技术: JMS (JavaEE 开发平台)

3 实验题目

Step1.不依赖于中间件概念（完全基于你的现有知识），完成一个软件设计（只要设计，不需要编码实现），实现 A 机器发送消息给 B 机器，并在 B 机器上显示的功能。此为第一版设计。

Step2.从中间件的角度，分析步骤 1 软件设计的优缺点，从架构、工作量、标准化、跨异构能力等角度提出若干改进点，并给出理由。完成改进版设计，此为第二版设计。

Step3.自学消息型中间件 Queue 和 Topic 的简单概念（大约 10-15 分钟），选择你认为合适的模型，改进步骤 2 的设计，并完成第三版设计（写出理由）。

Step4.学习使用 JMS (JavaEE 平台) 或 MSMQ (微软平台)（也可以选用其它中间件产品），设计开发一个即时通讯程序（重点在核心功能）。（除简单的文字通信功能外，要求实现包括但不限于传输文件、watchdog、存储转发、群聊等诸多 feature 中的一个或多个）

此处跳过 step1-3，直接提交 step4 结果

4 代码展示

实现思路已经体现在注释中

```
m pom.xml (Project3)  Main.java x
4 import org.apache.commons.io.FileUtils;
5
6 import javax.jms.*;
7 import java.io.BufferedReader;
8 import java.io.File;
9 import java.io.IOException;
10 import java.io.InputStreamReader;
11
12 public class Main {
13     //ActiveMq服务所在的地址和端口
14     //1个用法
15     private static final String BROKER_URL = "tcp://localhost:61616";
16     //用户订阅的TOPIC，只有相同的TOPIC才能收到相同的消息。（即群聊）
17     //1个用法
18     private static final String CHAT_TOPIC = "chatTopic";
19     //用户的用户名，用于唯一标识一个用户，后续也会作为ClientID使用
20     //3个用法
21     private static final String USER_NAME = "user1";
22
23     public static void main(String[] args) {
24         try {
25             //与ActiveMq建立连接
26             ConnectionFactory connectionFactory = new ActiveMQConnectionFactory(BROKER_URL);
27             Connection connection = connectionFactory.createConnection();
28             connection.setClientID(USER_NAME);
29             connection.start();
30
31             //设定消息确认模式（此处是自动确认）和TOPIC
32             Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
33             Topic topic = session.createTopic(CHAT_TOPIC);
34
35             //创建消息监听事件，便于收到其他用户消息
36             MessageConsumer consumer = session.createDurableSubscriber(topic, USER_NAME);
37             consumer.setMessageListener(message -> {
38                 if (message instanceof TextMessage) {
39                     //如果是TextMessage即为普通聊天消息
40                     try {
41                         String receivedMessage = ((TextMessage) message).getText();
42                         System.out.println("Received: " + receivedMessage);
43                     } catch (JMSException e) {
44                         e.printStackTrace();
45                     }
46                 } else if (message instanceof BytesMessage) {
47                     //如果是BytesMessage即为文本消息
48                     try {
49                         BytesMessage bytesMessage = (BytesMessage) message;
50                         byte[] fileBytes = new byte[(int) bytesMessage.getBodyLength()];
51                         bytesMessage.readBytes(fileBytes);
52
53                         // 指定保存文件的路径，此处以文本文件为例
54                         String filePath = "C:\\Users\\CC507\\Desktop\\output\\2.txt";
55                         FileUtils.writeByteArrayToFile(new File(filePath), fileBytes);
56
57                         System.out.println("File received and saved: " + filePath);
58                     } catch (JMSException | IOException e) {
59                         e.printStackTrace();
60                     }
61                 }
62             });
63         } catch (Exception e) {
64             e.printStackTrace();
65         }
66     }
67 }
```

```
m pom.xml (Project3)  Main.java x
61 //创建了MessageProducer对象，用于发送消息到主题
62 MessageProducer producer = session.createProducer(topic);
63
64 // 简单的命令行输入输出
65 BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
66 while (true) {
67     String messageText = reader.readLine();
68
69     if (messageText.startsWith("/file")) {
70         //如果以/file开头即为发送文件，后跟文件路径即可
71         String filePath = messageText.substring(6); // 提取文件路径
72
73         File file = new File(filePath);
74         if (!file.exists() || !file.isFile()) {
75             System.out.println("Invalid file path!");
76             continue; // 如果文件不存在或者不是一个普通文件，则继续下一次循环
77         }
78
79         byte[] fileBytes = FileUtils.readFileToByteArray(file); // 读取文件内容
80
81         BytesMessage bytesMessage = session.createBytesMessage();
82         bytesMessage.writeBytes(fileBytes);
83         producer.send(bytesMessage); // 发送字节消息
84     } else {
85         //发送普通消息
86         TextMessage textMessage = session.createTextMessage("s: USER_NAME + ": " + messageText);
87         producer.send(textMessage); // 发送文本消息
88     }
89 }
90
91 } catch (JMSException | IOException e) {
92     e.printStackTrace();
93 }
94 }
95
96 }
```

5 实验结果

（命令行的红字是没有设置日志实现所致，并不影响运行）

（发送方和接收方的代码一致，但是需要保证 USER_NAME 不同，两者导致仍然可以运行）

（需要保证 ActiveMq 服务正在运行，此处在本机上运行，实际投入生产需要更改到正确的地址）

功能 1：群聊。

所有订阅了相同 TOPIC 的用户都将收到相同的消息

发送方:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\JetBrains\IntelliJ IDEA 2023.2.5\lib\idea_rt.jar" -Dfile.encoding=UTF-8
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
123456hello
Received: user1: 123456hello
```

接收方: (这里只以一个用户为例, 实际上多个接受方收到的也相同)

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\JetBrains\IntelliJ IDEA 2023.2.5\lib\idea_rt.jar" -Dfile.encoding=UTF-8
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Received: user1: 123456hello
```

功能 2: 存储转发。

我使用了持久订阅者, 意味着消息在 ActiveMq 上将会以队列的形式组织, 并存储起来。存储后, 中间节点会负责将消息从发送方传递到接收方。这个过程可以包括消息的路由、传输和最终投递给接收方。

```
//创建消息监听事件, 便于收到其他用户消息
MessageConsumer consumer = session.createDurableSubscriber(topic, USER_NAME);
```

具体的实现效果是: 将接收方的程序关闭, 发送方发送一个消息, 再启动接收方的程序, 仍然可以收到消息

```
运行 Main x
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\JetBrains\IntelliJ IDEA 2023.2.5\lib\idea_rt.jar" -Dfile.encoding=UTF-8
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Received: user1: 77789
```

功能 3：文件传输。

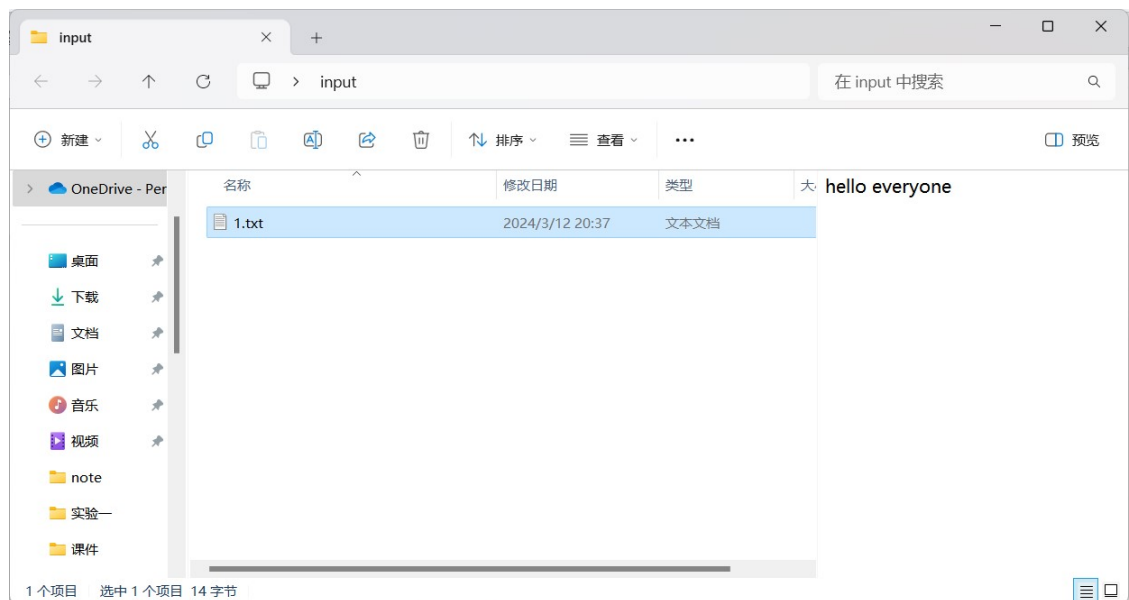
我使用字节消息和文本消息来区分文件消息和普通消息，当收到字节消息的时候以文本方式处理并存储。

此处我将路径指定在代码文件中，实际投入生产可以更改使之更加灵活。

发送方：



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\JetBrains\IntelliJ IDEA 2023.
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
123456hello
Received: user1: 123456hello
77789
Received: user1: 77789
/file C:\Users\CC507\Desktop\input\1.txt
File received and saved: C:\Users\CC507\Desktop\output\2.txt
|
```



接收方：

```
运行 Main x
" C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\JetBrains\IntelliJ IDEA 2023.2.5\lib\ide
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Received: user1: 77789
File received and saved: C:\Users\CC507\Desktop\output\2.txt
```

