DEZSYS06

JMS Chat

von René Hollander und Patrick Malik 4AHIT

22.11.14

Table of Contents

Angabe

Implementieren Sie eine Chatapplikation mit Hilfe des Java Message Service. Verwenden Sie Apache ActiveMQ (http://activemq.apache.org) als Message Broker Ihrer Applikation. Das Programm soll folgende Funktionen beinhalten:

Benutzer meldet sich mit einem Benutzernamen und dem Namen des Chatrooms an.
Beispiel für einen Aufruf:

vsdbchat <ip_message_broker> <benutzername> <chatroom>

Der Benutzer kann in dem Chatroom (JMS Topic) Nachrichten an alle Teilnehmer eine Nachricht senden und empfangen.
Die Nachricht erscheint in folgendem Format:

<benutzername> [<ip_des_benutzers>]: <Nachricht>

Zusätzlich zu dem Chatroom kann jedem Benutzer eine Nachricht in einem persönlichen Postfach (JMS Queue) hinterlassen werden. Der Name des Postfachs ist die IP Adresse des Benutzers (Eindeutigkeit).

Nachricht an das Postfach senden:
MAIL <ip_des_benutzers> <nachricht>

Eignes Postfach abfragen:
MAILBOX
Der Chatraum wird mit den Schlüsselwort EXIT verlassen. Der Benutzer verlaesst den Chatraum, die anderen Teilnehmer sind davon nicht betroffen.
Gruppenarbeit: Die Arbeit ist in einer 2er-Gruppe zu lösen und über das Netzwerk zu testen! Abnahmen, die nur auf localhost basieren sind unzulässig und werden mit 6 Minuspunkten benotet!

Software:

Apache ActiveMQ Installationspaket ist unter Resource verfügbar.

Quellcode:

jmschat.jar

Benotungskriterien:

o 2 Punkte: Installation Message Broker Apache ActiveMQ
o 8 Punkte: Implementierung des Chatraums (JMS Topic)
o 6 Punkte: Implementierung der Postfach-Funktionalität (JMS Queue)

Quellen:

http://activemq.apache.org/index.html
http://www.academictutorials.com/jms/jms-introduction.asp
http://docs.oracle.com/javaee/1.4/tutorial/doc/JMS.html#wp84181
http://www.openlogic.com/wazi/bid/188010/How-to-Get-Started-with-ActiveMQ
http://jmsexample.zcage.com/index2.html

http://www.onjava.com/pub/a/onjava/excerpt/jms_ch2/index.html
http://www.oracle.com/technetwork/systems/middleware/jms-basics-jsp-135286.html
http://java.sun.com/developer/technicalArticles/Ecommerce/jms

Designüberlegung

- Message Objekt:

Enthält folgende Attribute:
  - IP des Absenders
  - Nickname des Absenders
  - Inhalt in Form eines String
- Chatroom:

Der Chatroom wird über ein JMS Topic gelöst. Gesendet wird ein serialisiertes Message Objekt. Mit dem CHATROOM Kommando kann man den Chatroom wechseln. Mit dem EXIT Kommando verlässt man den Chat und das Programm schließt sich.

- Mailbox:

Die Mailbox wird über eine JMS Queue gelöst. In die Queue kann man Nachrichten schicken die am Server gespeichert werden. Über das MAILBOX Kommando kann der Benutzer seine Nachrichten abrufen. Sie werden dann vom Server gelöscht. Über das MAIL Kommando kann man eine Nachricht an einen anderen Nutzer schicken. Zur identifizierung wird nur der Username verwendet.

Zeitaufstellung

Zeitschätzung

600 Minuten Implementieren und Kommentieren
30 Minuten Protokoll pro Person

Zeitaufzeichnung

Rene Hollander	Implementation von JMSChat, Chatroom und Mailbox	250 Minuten
Rene Hollander	Bugfixing	50 Minuten
Rene Hollander	Protokoll	30 Minuten
Patrick Malik	Implementierung der CLI	120 Minuten
Patrick Malik	Testen	120 Minuten

Name	Geschätzt	Tatsächlich
René Hollander	430 Minuten	330 Minuten
Patrick Malik	200 Minuten	240 Minuten

Geschätzt	Tatsächlich
630 Minuten	570 Minuten

Arbeitsdurchführung/Lessons Learned

- Um die Nachrichten zu versenden wird die Java Serialisierungs API verwendet. ChatMessage wird in ein byte[] serialisiert und als JMS BytesMessage verschickt.
- Der Aufgabenstellugn haben wir noch den Befehl “CHATROOM <chatroomname>”

hinzugefügt um während der Laufzeit flexibler zu sein und auch noch heir den chatroom wechseln zu können.
- Ein Thread wartet auf das ankommen einer Nachricht und schreibt diese beim ankommen in die Konsole.
- Die Interaktion mit der Kommandozeile wurde simple mit BufferedReadern und Scannern gehandhabt. Zudem wurden direkt beim entgegennehmen der Befehle Exceptions gehandelt.
- Der Output wird von Loggern durchgeführt.
- Die IP eigene IP wurde online abgerufen.

Testbericht

- Da wir leider nicht wissen wie Userinput beim Testen simuliert warden soll, mussten wir das leider außen vor lassen. Sämtliche Versuche den UserInput zu simulieren sind gescheitert (siehe Code)
- Interessanterweise gibt es einen Testcase der alleine getsten erfolgeich ist, allerdings beim Durchlauf aller testcases eine Excpetion wirft, weshalb ist nicht bekannt.

Abgabe auf Github

Link: https://github.com/ReneHollander/dezsys06_jmschat