Melanie Göbel , Tobias Perny

JMS

Chatapplikation mit Java Message Service

Inhalt

Aufgabenstellung

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.

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)
Abnahmen, die nur auf localhost basieren sind unzulässig und werden mit 6 Minuspunkten benotet!

Arbeitsaufteilung

Aufwandsabschätzung

Aufgabe	Person(en)	Zeitschätzung in Minuten
Installation von Apache ActiveMQ	Göbel, Perny	6
Implementierung Chatroom	Göbel	120
Implementierung Postfach	Perny	120
Testen	Göbel,Perny	30
Protokoll	Göbel	40
Bugfixing	Perny	60
Gesamt	Göbel	178 = 2 Stunden 58 Min
Gesamt	Perny	198 = 3 Stunden 18 Min
Gesamt	Gesamt	376

Endzeitaufteilung

Aufgabe	Person(en)	Zeitschätzung in Minuten
Installation von Apache ActiveMQ	Göbel, Perny	6
Implementierung Chatroom + Menu	Göbel	150
Implementierung Postfach	Perny	70
Testen	Göbel,Perny	20
Protokoll	Göbel	40
Bugfixing	Perny	120
Gesamt	Göbel	203 = 3 Stunden 22 Min
Gesamt	Perny	203 = 3 Stunden 22 Min
Gesamt	Gesamt	406

Theorie

Java Message Service

Ist eine Java-API die Nachrichten via eine Message Oriented Middleware (MOM) Nachrichten zwischen Sender und Receiver vermittelt. Die Message Oriented Middleware verwaltet die Nachrichten mit 2 verschiedenen Konzepten

Queue (Point-to-Point)
Jede Nachricht bekommt einen bestimmten Nachrichtkanal und wird zur Abholung bereitgestellt. Nachrichten können nur einmal abgeholt werden, das System folgt nach den *First Come First Serve* Prinzip.

Topic (Publisher/Subscriber)
Ein Absender schickt eine Nachricht an mehrere Empfänger (Subscriber), die Nachricht wird automatisch ausgeteilt. Wenn sich ein Empfänger abmeldet, hat es keine Auswirkung auf andere Empfänger.

Designüberlegung

Wir benötigen eine Klasse zum Starten der Anwendung sowie für das Menü außerhalb und innerhalb des Chats. Nach dem Aufruf wird nach der IP-Adresse von den Active-MQ sowie Benutzername gefragt.

Chat.java
Schreibt und sendet alle Nachrichten in einem Topic, das Menü ist nicht dabei (sondern in Start.java)

Mail.java
Schreibt Mails und ruft sie ab, indem eine Queue erstellt wird mit den Namen der IP-Adresse des Senders.

Hilfklassen
Zur Überprüfung von der IP-Adresse sowie das Format der IP-Adressen von den Empfänger benötigen wir eine Hilfsklasse.

Arbeitsdurchführung

Eingaben überprüfen

Bevor man einen Chat implementiert muss man sich um Eingaben kümmern, das Problem dabei ist zu Überprüfen ob sie auch gültig sind. Mit Hilfe von regulären Ausdrücken, kann man überprüfen ob eine IP-Adresse, sowie Benutzername gültig sind.

IP-Adresse[1]: `(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)`

Beispiel	Match
192.168.119.15	True
8.8.8.8	True
10.a.134.12	False
192.168.3	False

Benutzername[2]: `^[a-z A-Z 0-9_-]{3,15}$`

Beispiel	Match
Melanie	True
tobi_19	True
Göbel	False
t0b!as	False

Menü

Menü ist solange da, bis ein Das Chat eröffnet wird oder exit eingegeben wird.
Im Menü gibt es mehrere Befehle (mailbox, mail, chat, exit)
Gelöst wurde es mit einen switch(String), bei exit wurde System.exit(0); durchgeführt.

Wenn zb wie beim chat dannach mehrere Argumente stehen, muss jede Eingabe gesplittet werden. Dadurch bekommt man nach dem splitten alle wichtigen Informationen für den jeweiligen Befehl.

Chat

Während dem Chat kann man ebenfalls Mails schreiben, dazu benötigt man ein zweites Menü, damit man jedoch mail auch als Nachricht schreiben kann sind alle Argumente mit einen / vorher.
Mit exit kommt man wieder ins normale Menü.

CreateTopic(): Erstellt ein Topik mit den Namen des Chatrooms (im Menü mit angegeben)
OnMessage(Message): Sendet die Nachricht
sendTopicMessage(String): hängt benutzername[ip]: an jede Nachricht wenn es nicht darum geht ob jemand online oder offline ist (Lösung mittels regulären Ausdrücken)
stopChat(): Schließt alles.

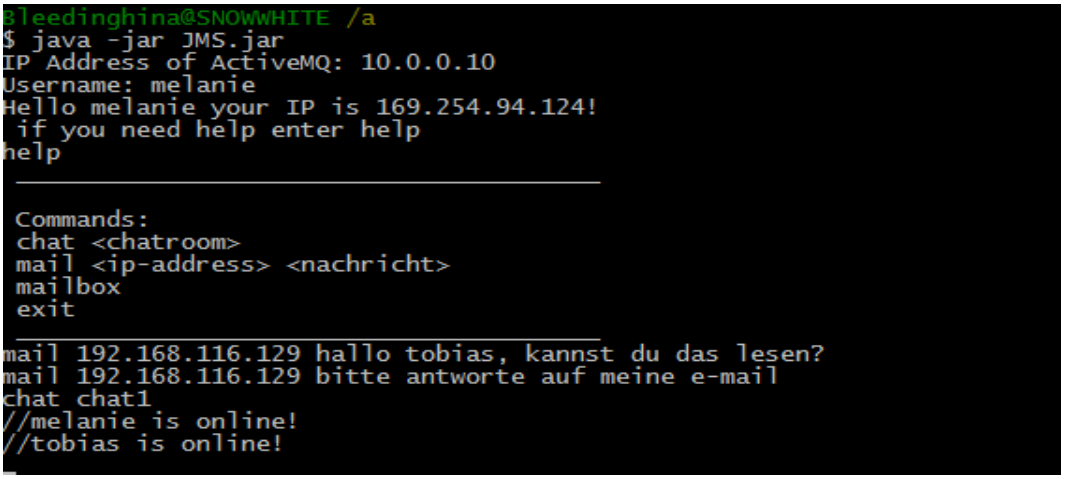
Mail

Bevor überhaupt eine E-Mail geschrieben wird, muss man die Queue zum Empfangen und Senden öffnen. Dies soll gleich passieren, nachdem man den Namen weiß (IP-Adresse). Durch die IP-Adresse ist es eindeutig an wen die Nachricht geht, die Nachricht kann beliebig lang sein. Problem anfangs: wir haben den String mit .split(„ „); nach Leerzeichen getrennt und als Nachricht das dritte String im Array verwendet. Bessere Lösung: Man zählt nach wieviele Stellen das Array hat und zieht die ersten 2 für das wort mail und die IP-Adresse ab.

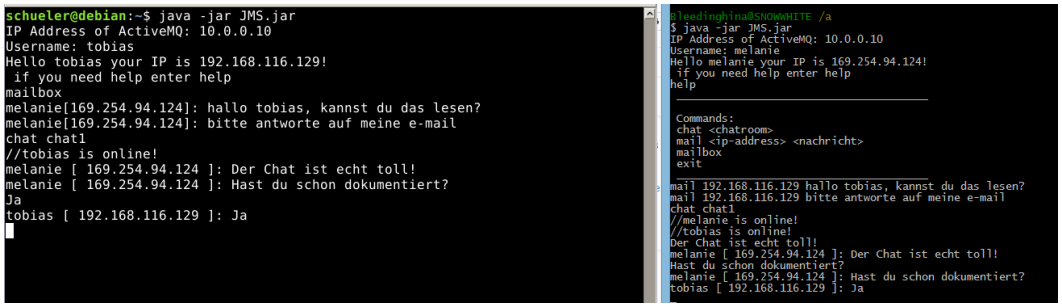
startMailbox(): Startet die Mailbox, muss gleich am Anfang getan werden
readMails(): liest in der Queue ob es Nachrichten gibt
writeMail(String, String): schreiben einer E-Mail an die IP-Adresse
onMessage(): Sendet die Nachricht

Testbericht

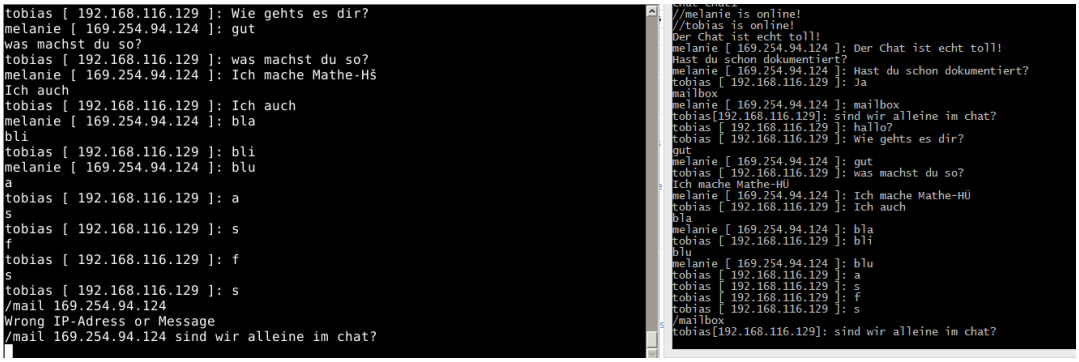
Anmelden, Hilfe aufrufen und sehen wer online ist funktioniert:



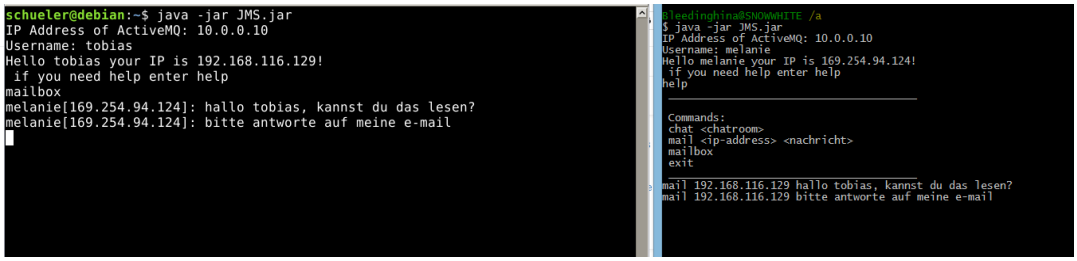
Chatten funktioniert:



Mail während den chatten senden und abrufen funktioniert:



Mail funktioniert ebenso außerhalb des Chats:



Quellenangaben

[1] IP-Adresse auf Richtigkeit überprüfen (RegEx) – subjective
http://www.webwork-community.net/posting7967_23_0.html (last seen: 23.11.2014)

[2] How to validate username with regular expression - mkyong
http://www.mkyong.com/regular-expressions/how-to-validate-username-with-regular-expression/
(last seen: 23.11.2014)