JMS-Chat

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

Organisation

Zeitabschätzung

	Designüberlegung + UML	Implementierung	Dokumentation
Ableitinger	1h	3h	0,5h
Hackenberger	1h	2h	1,5h

Tatsächlicher Zeitaufwand

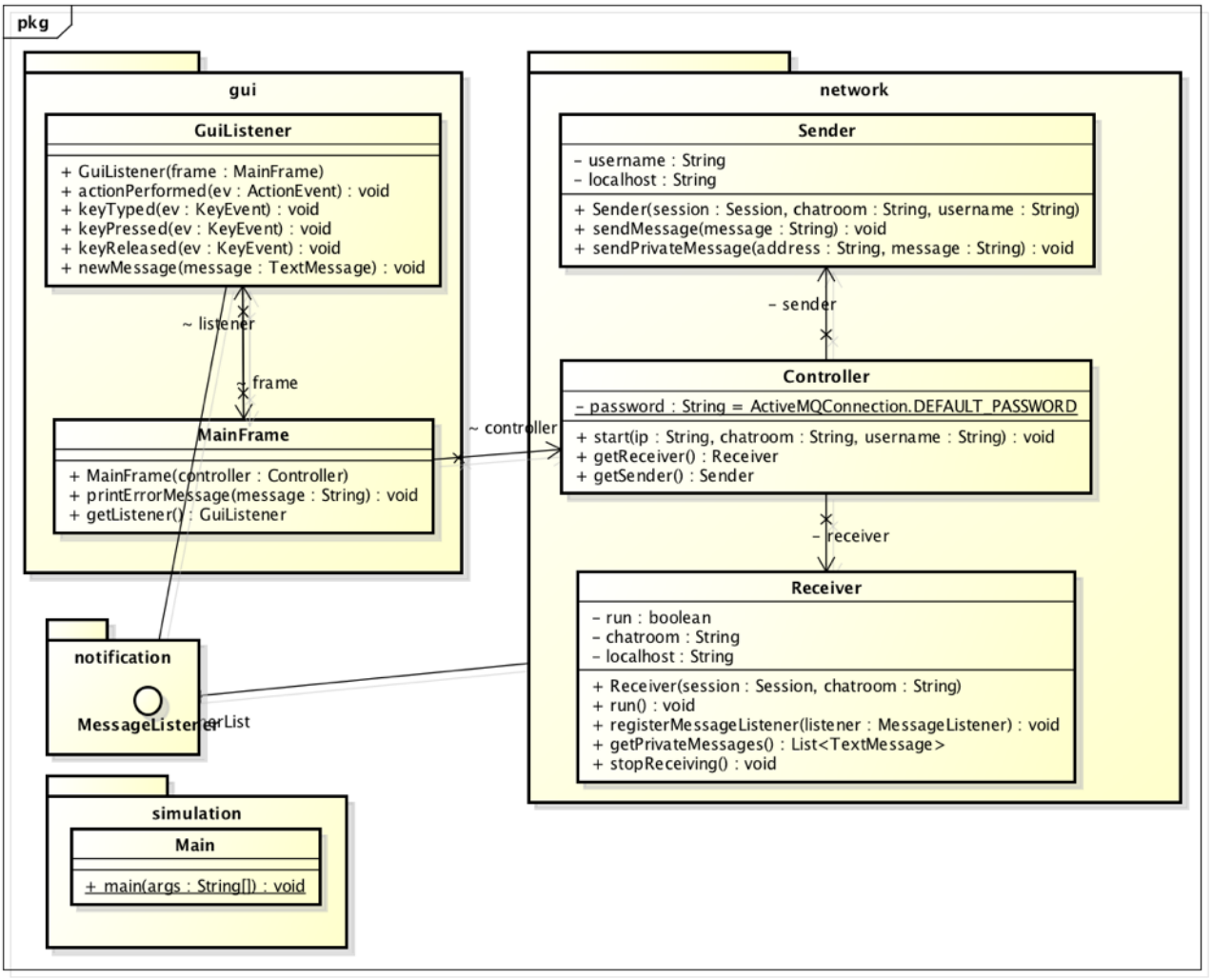
	Designüberlegung + UML	Implementierung	Dokumentation
Ableitinger	0,5h	4h	1h
Hackenberger	0,5h	3h	1h

Arbeitseinteilung

Hackenberger	Ableitinger
Implementierung der Sender und Controller Klasse	Implementierung der Receiver Klasse
Implementierung der Main Klasse	Implementierung der GUI
Dokumentation	Dokumentation

Design

UML



Das Grundlegende Design besteht darin, dass es einen Sender und einen Receiver gibt, welche über den Controller verwaltet werden.

bei der Initialisierung wir dem MainFrame der Controller übergeben, welches sich um alle weiteren Aktionen kümmert.

Wenn nun in der GUI eine Nachricht versendet wird, wird diese and den Sender weitergeleitet und versendet. Während der Receiver als HintergrundThread permanent neue Nachrichten abruft.
Der GUIListener wird über den MessageListener, welcher am Receiver registriert werden kann, benachrichtigt wenn eine neue Nachricht kommt und stellt sie dar.

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