3 Asynchrone Middleware JMS

3.1 Zweck

Die ersten Erfahrungen mit JMS und Messaging zu sammeln und die beiden Topologien Point to Point und Publish/Subscribe in der Praxis kennen und beobachten zu lernen.

Beantworten sie alle in *Kursivschrift blau geschriebenen* Fragen in einer Datei und laden sie diese bei der https://moodle.ost.ch/mod/assign/view.php?id=331738 (Übung 3. Abend) als **PDF** hoch

3.2 System-Umgebung

Sie arbeiten mit den Werkzeugen Eclipse und Java Development Kit 1.11.x. oder höher. Bis OpenJDK 11 wurde die Übung getestet.

Die Übungs-Programm-Vorlagen können von der Übungsseite 3. Abend Moodle als `CDSUebung_3-1.*` als zip oder tar vom Ordner *Skripte und Übungen* geladen, kopiert und ausgepackt werden. Sie können das entsprechende Eclipse-Projekt mit Import als gradle-Projekt einlesen.

> ***Wichtiger Hinweis***: Bitte darauf achten, dass im Pfad der auf das Projekt zeigt, keine Leerschläge (Blanks) und Sonderzeichen (wie Umlaute etc) enthalten sind.

Für Uebung 3 brauchen sie einen JMS Server. Wir verwenden als JMS-Provider ActiveMQ V5.14.3, ein OpenSource-Produkt.

Für diese Übung muss aber nichts installiert werden, da in einer Cloud der Message-Provider (AMQ) bereitgestellt wird und sie so auch auf die Queues/Topics ihrer Kollegen:innen zugreifen können

Man kann das von Apache herunterladen:
https://activemq.apache.org/components/classic/download/
Die Installation wird im Wesentlichen mit dem Entpacken des Zip- oder tar-Files gemacht und dann gemäss der Dokumentation den Message-Provider starten.

Oder man installiert den ActiveMQ mit docker-compose wie unter ./docker/README.md beschrieben.

Die Provider-Adresse muss bei den verschiedenen Übungsteile mit Parameter `-host <MQHost>` (setzen sie `mq-mas-se.cloud.oeint.ch` anstelle von `<MQHost>` ein) eingegeben werden. Das Default-Port des Message-Providers ist 61616 muss nicht eingegeben werden.

Mit der Admin-Konsole kann man die aktuell angelegten Queues und Topics einsehen und auch die Anzahl Messages sowie Inhalt der Queues ansehen. Man erreicht die Admin-Konsole des Message-Provders mit dem Browser unter http://mq-mas-se.cloud.oeint.ch:8161/admin. Login/Pwd ist admin/cds4admin*

wobei in den Tabs (Register) das folgende angezeigt werden kann:

- Queue: Anzeigen der Queues
- Topics : Anzeigen der Topics für Publish/Subscribe

3.3 Problemstellung

Verschiedene Szenarien von Point to Point- und Publish/Subscribe-Topologien zu beobachten und zu untersuchen und daraus Schlüsse zu ziehen.

3.4 Aufgabenstellung

3.4.1 Studieren der Programme

Folgende Files sind kurz zum besseren Verständnis beschrieben:

(Basispfad = ./src/main/java/ch/ost/mas/cds/jmstutorial)

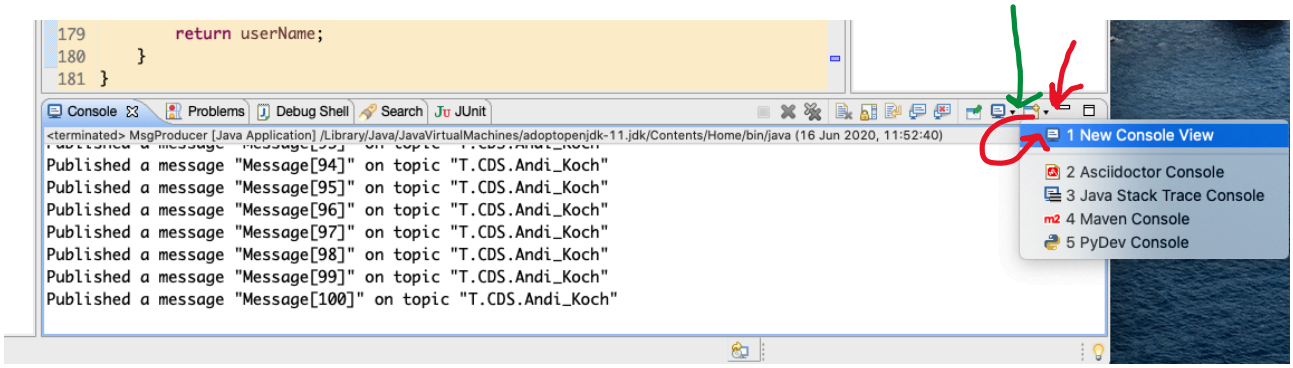
ptp/MsgQueueReceiver.java	Für Point to Point Messaging, stellt dieses Programm den Meldungsempfänger (Consumer) dar.
ptp/MsgQueueSender.java	Für Point to Point Messaging, stellt dieses Programm den Sender (Producer) dar.
ptp/MsgQueueReplyer.java	Für Point to Point Messaging, stellt dieses Programm den Meldungsempfänger (Consumer) dar. Im Gegensatz zu MsgQueueReceiver verwendet dieses Programm die mitgelieferte Absender-Queue-Adresse, um eine Antwort zurückzusenden (quasi Client/Server).
ptp/SyncMsgQueueSender.java	Für Point to Point Messaging, stellt dieses Programm den Sender (Producer) dar. Im Gegensatz zu MsgQueueSender sendet dieses Programm die Adresse seiner selbst erstellten temporären Queue mit, wohin der Empfänger seine Antwort senden kann. Dieses Programm wartet, dann (zwar mit einem Timeout) bis die Antwort auf seine Anfrage zurückkommt. (quasi Client/Server)
ps/MsgConsumer.java	Für Publish/Subscribe Messaging, stellt dieses Programm den Subscriber (Consumer) dar.
ps/MsgProducer.java	Für Publish/Subscribe Messaging, stellt dieses Programm den Publisher (Producer) dar.
util/JMSUtil.java	Eine Hilfsklasse, welche verschiedene Funktionen zum Initialisieren des JMS-Servers bewerkstelligt.

Wichtig: Dokumenation:	Die JMS-Online-Dokumentation zu finden unter https://docs.oracle.com/cd/E17802_01/products/products/jms/javadoc-102a/index.html

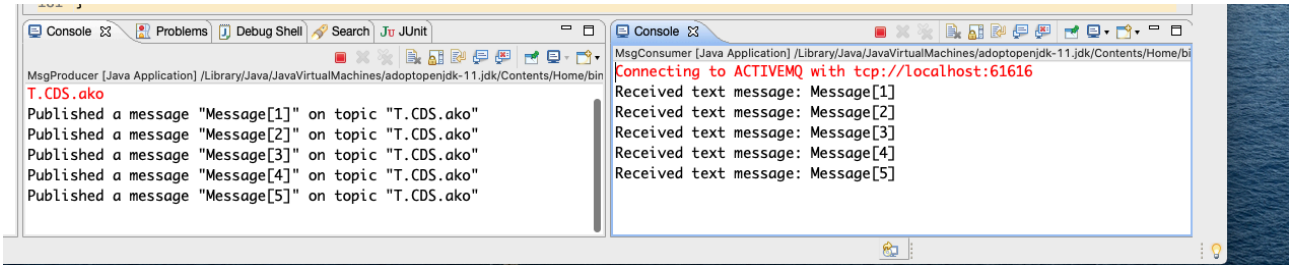
3.4.1.1 Tip für Eclipse Consolen-Ausgabe

Bei der Übung kann man ja verschiedene Prozesse gleichzeitig laufen lassen. Damit man deren Output gut beobachten kann, ist es möglich mehr als ein Console-Fenster anzuzeigen (pro Programm):

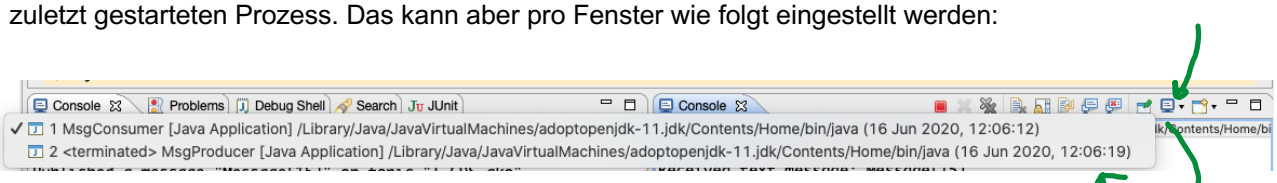
Man öffnet über das Menu (siehe Bild) ein weiteres Konsolen-Fenster. Das kann man dann mit Drage&Drop seitlich von den anderen Konsolenfenster anordnen.



Das sieht dann z.B. so aus:



Links sieht man dann den *Publisher* rechts den *Subscriber*. Am Anfang *hören* dann beide Fenster auf den zuletzt gestarteten Prozess. Das kann aber pro Fenster wie folgt eingestellt werden:



Wenn den Parameter ***-count n*** nicht angibt, dann werden 100 Meldungen produziert und konsumiert. Passen sie die Parameter so an, dass es für die jeweilige Aufgabe passt. Zum Beispiel, kann es sein, dass der Empfänger mehr Meldungen empfangen muss, wenn mehrere Sender aktiv sind.

Hinweis: *Wenn man einen Prozess abbricht, kann es sein, dass noch Meldungen in der Queue nicht abgeholt wurden. Das kann ein Verhalten zeigen, das man es logisch nicht nachvollziehen kann. Man muss dann u.U. die Meldungen in einer Queue mit der Admin-Konsole von ActiveMQ löschen.*

3.4.2 Point to Point Messaging

Wichtig: Für alle Übungen mit ActiveMQ benötigen sie keine weiteren Einstellungen. ActiveMQ wurde als Default-Einstellung gesetzt. Für andere Message-Provider einzusetzen, schauen sie bitte im JMSUtil.java nach. Die Parameter bei den jeweiligen Teilübungen geben sie bitte bei "Arguments" unter Debug... an. sie müssen aber den Hostnamen des Messageproviders z.B. -host `<MQHost>` (default ist **localhost**) bei jedem der gestarteten Programme angeben. Ersetzen sie `<MQHost>` mit mq-mas-se.cloud.oeint.ch

Der Name der Queue ist jeweils `Q.CDS.<loginname>` und die des verwendeten Topic ist `T.CDS.<loginname>` z.B. für *rmeier* `Q.CDS.rmeier` resp. `T.CDS.rmeier`. So können sie auch mal eine Mitteilung einem Ihrer(r) Kollegen|innen senden.

Mit dem Browser können sie dann unter **`http://mq-mas-se.cloud.oeint.ch:8161/admin`** die Adminkonsole verwenden.

3.4.2.1 Producer/Consumer (asynchron)

- Starten sie den Sender MsgQueueSender mit dem Parameter `-count 20 -host <MQHost>`
- Starten sie den Empfänger MsgQueueReceiver mit dem Parameter `-count 20 -host <MQHost>`
- Beobachten sie ab welcher Meldungsnummer der Empfänger Meldungen empfängt.

Generell: Zeichnen sie die Test-Konstellation auf Papier (oder elektronisch) auf. Speichern sie den Consolen-Output in ein Textfile auf.

3.4.2.2 Producer/Consumer (zeitversetzt)

- Wiederholen sie 3.4.2.2 aber warten sie mit Schritt 2) solange, bis der Sender fertig ist.
- Starten sie den Empfänger und beobachten sie ab welcher Meldungsnummer der Empfänger die Meldungen empfängt.

3.4.2.3 Ein Sender, mehrere Empfänger

- Starten sie den Sender MsgQueueSender mit -count 30 -host <MQHost>
- Starten sie zwei Receiver MsgQueueReceiver
- *Beobachten sie, wieviele Meldungen jeder Empfänger erhalten hat (Vergleichen sie die Meldungnummern).*
- Zeichnen sie die Test-Konstellation auf.

3.4.2.4 Synchronisierter Meldungsaustausch (Client/Server)

- Starten sie den synchronisierenden Sender SyncMsgQueueSender -count 10 -host <MQHost>
- Starten sie den Receiver MsgQueueReplyer, welcher auf jede Meldung eine Antwort gibt.
- *Beobachten sie, was beide Programme ausgeben und ziehen sie ihre Schlüsse.*
- Zeichnen sie die Test-Konstellation auf.

- Schauen sie sich den Programmcode an.

3.4.2.5 Synchronisierter Empfänger mit mehreren Sendern

- Starten sie einen Receiver MsgQueueReplyer, welcher auf jede Meldung eine Antwort gibt.
- Starten sie zwei bis drei synchronisierende Sender SyncMsgQueueSender.
- *Beobachten sie darauf, welcher Sender welche Meldung (Temporäre Queue Nr) sendet, und ob der Empfänger auch dem richtigen Sender die Antwort zurück sendet.*
- Zeichnen sie die Test-Konstellation auf.

3.4.3 Publish/Subscribe

3.4.3.1 Einfacher Publisher mit Subscriber (1:1)

- Starten sie einen Publisher MsgProducer mit -count 15 -host <MQHost> auf.
- Starten sie einen Subscriber MsgConsumer auf.
- Zeichnen sie die Test-Konstellation auf.
- *Beobachten sie, ab welcher Meldungsnummer der Consumer die Meldungen erhält.*
- Versuchen sie es auch mit mehreren Publishers

3.4.3.2 Einfacher Publisher mit mehreren Subscribern (1:n)

- Starten sie einen Publisher MsgProducer mit -count 100 -host <MQHost> auf.
- Starten sie mehrere (3-4) Subscriber MsgConsumer etwas zeitversetzt auf.
- Zeichnen sie die Test-Konstellation auf.
- *Beobachten sie, ab welcher Meldungsnummer die Consumer die Meldungen erhalten*

3.4.3.3 Mehrere Publisher mit einem Subscriber (n:1)

- Wie 3.4.3.1 aber Publisher mehrmals gestartet und *beobachten sie den Subscriber*.

3.4.4 Konzept der Programme untersuchen

- Schauen sie sich die Programme in der Reihenfolge der obigen Tests an. Lassen sie dabei entweder den Sender oder Empfänger (resp. Publisher oder Consumer) im Debug schrittweise durchlaufen und schauen, was so passiert. Dabei geht es um das Verständnis, was wo passiert.
- Erstellen sie selber ein Programm (nehmen sie eines der Programme als Vorlage), welches von ihrer Queue eine Meldung empfängt und sie an eine andere Queue weiterleitet. Ein Programm, welches auf dieser Queue hört, soll dann eine Antwort an den urspünglichen Sendern zurück senden. Welchen JMS-Mechanismus können sie verwenden?

3.4.4.1 Schlussfolgerung und Zusammenfassung

Erstellen sie eine kurze Zusammenfassung aller oben markierten Beobachtungen in einem PDF-Document und schreiben sie auf, was sie daraus schliessen.

Schreiben sie je drei praktische Beispiele aus Ihrem Berufsumfeld auf, bei denen sie die eine oder andere (PTP oder P/S) Topologie einsetzen könnten.

Viel Erfolg!

16.06.2020/ako (online)
10.05.2021/ako
24.4.2022/ako
4.05.2023/ako