Hochschule Darmstadt Fachbereich Informatik J. Reichardt

## ANLEITUNG ZUM 3. PRAKTIKUMSVERSUCH VERTEILTE SYSTEME

# **Publisher-Subscriber**

mit JMS und Message Driven Beans

## TEIL A: TaskBag-Topic mit Task-Producer und Task-Consumer erstellen

#### A.1 Task-Consumer erstellen

#### • Projektdateien und -verzeichnisse erzeugen

• File -> New Project (*beachten: Java EE und EJB Module auswählen*)

🗊 New Project		X
Steps	Choose Project	
1. Choose Project 2	Categories: Java Ueb Java EE Java Card Java ME Maven Groovy NetBeans Modules B Samples	Projects: Chterprise Application Enterprise Application with Existing Sources EIB Module with Existing Sources Enterprise Application Client Enterprise Application Client with Existing Sources Packaged Archive
	Description:	
	Creates a new Enterprise JavaB Standard projects use an IDE-gene project.	ean (EJB) module in a standard IDE project. rated Ant build script to build and run your
	< Back	Next > Enish Cancel Help

#### o Next

Steps	Name and Location	
1. Choose Project	Project Name: TaskConsumer	
<ol> <li>Server and Settings</li> </ol>	Project Location: umente und Einstellungen\Apollo\Eigene Dateien\NetBeansProjects	Browse
	Project Eolder: jinstellungen\Apollo\Eigene Dateien\NetBeansProjects\TaskConsumer	
	Use Dedicated Folder for Storing Libraries	
	Libraries Folder:	
	Different users and projects can share the same compilation libraries (see Help for details).	
	Set as Main Project	

o Next

🗊 New EJB Module	X
Steps	Server and Settings
1. Choose Project 2. Name and Location	Add to Enterprise Application:
3. Server and Settings	Server: GlassFish Server 3
	Lise dedicated ibrary folder for server JAR files
	Java EE Version: Java EE 6 💌
	Enable Contexts and Dependency Injection
	¢
	< Back Next > Einish Cancel Help

- o Finish
- Folgende Projektdateien und -verzeichnisse wurden erzeugt:



- Message Driven Bean erzeugen
  - o Rechts-Click auf Ordner ,Source Packages/<default package>' -> New -> Message Driven Bean

🇊 New Message-Driven Bean	X
Steps	Name and Location
<ol> <li>Choose File Type</li> <li>Name and Location</li> </ol>	E38 Name: TaskConsumerMD8
	Project: TaskConsumer
	Location: Source Packages
	Package: taskconsumer
	Project Destinations:
	C Server Destinations: jms/XTopic
	Valid message destination must be selected.
	Kenter State Kenter Ken

• Add Project Destination (*nur falls die Topic noch nicht existiert*. Anderenfalls, z.B. wenn die Topic von anderen Consumern bereits benutzt wird: Topic unter ,Server Destinations' auswählen)

🗊 Add Message	Destination	2
Destination Name:	jms/TaskBagTopic	
Destination <u>Type</u> :	C Queue	
	• Topic	
	OK Cancel Help	

o OK

🗊 New Message-Driven Bean	×
Steps	Name and Location
<ol> <li>Choose File Type</li> <li>Name and Location</li> </ol>	EJB Name: TaskConsumerMDB
	Project: TaskConsumer
	Location: Source Packages
	Package: taskconsumer
	Project Destinations: jms/TaskBagTopic     Add
	C Server Destinations: jms/XTopic
The second	
	<back next=""> Enish Cancel Help</back>

- o Finish
- Im Verzeichnis 'Source Packages/taskconsumer' wurde die Datei TaskConsumerMDB.java angelegt, mit folgendem Inhalt:

```
@MessageDriven(mappedName = "jms/TaskBagTopic", activationConfig =
{
    @ActivationConfigProperty(propertyName = "acknowledgeMode",
            propertyValue = "Auto-acknowledge"),
     @ActivationConfigProperty(propertyName = "destinationType",
     propertyValue = "javax.jms.Topic"),
@ActivationConfigProperty(propertyName = "subscription
            Durability", propertyValue = "Durable"),
     @ActivationConfigProperty(propertyName = "clientId",
            propertyValue = "TaskConsumerMDB"),
     @ActivationConfigProperty(propertyName = "subscriptionName",
            propertyValue = "TaskConsumerMDB") })
public class TaskConsumerMDB implements MessageListener {
 public TaskConsumerMDB() {
 }
 public void onMessage(Message message) {
 }
}
```

- Die Methode *onMessage(Message)* ausprogrammieren:
  - Message casten nach TextMessage.
  - Zugriff auf Text mittels getText().
  - JMSException abfangen.
- TaskConsumer bereitstellen
  - Rechts-Click auf Ordner 'TaskConsumer' -> Deploy.
- Admin-Konsole starten
  - o Rechts-Click auf ,Services/Servers/GlassFish Server 3' und ,View Admin Console' wählen.



• Auf der Admin-Konsole ist unter ,Resources/JMS Resources' die während des Deploy-Vorgangs erzeugte TaskBag-Topic mit ihrem globalen JNDI-Namen zu sehen.

J <b>I</b> JMS	IS Destination Resources	k New to create a new des	stination resource. Click the name
	,		
Des	tination Resources (1)		
Des E	tination Resources (1) B New Delete Enable Disable		<b>D T</b>
Des EV (	tination Resources (1) B New Delete Enable Disable JNDI Name	Enabled	Resource Type

 Auf der Admin-Konsole ist unter ,Configuration/Java Message Service/Physical Destinations/Statistics/View' eine Zustandsanzeige der Topic zu erhalten (Number of Consumers, Producers, Messages, Bytes, ...).

2	JMS Physical Destinations				
	Java Message Senice (JMS) physical destination objects are maintained by Message Queue brokers. The queue named mq.sys.dmq is the system destin messages are redirected. Click New to create a new physical destination.				
	Destinations (2) New Delete Flush	_			
	Name	Ť4	Туре	Ť4	Statistics
	🗖 mq.sys.dmq		queue		View
	PhysicalTopic		topic		View

 Bei Überlauf der Topic (Fehlermeldung "Topic akzeptiert keine weiteren Nachrichten"), beispielsweise durch Nichtauslieferung von Nachrichten aufgrund des Message Selektors in Teil B, kann die Topic mithilfe der *Flush*-Funktion wieder geleert werden.

## A.2 Task-Producer erstellen

- Projektdateien und -verzeichnisse erzeugen
  - o File -> New Project (beachten: Java EE und Enterprise Application Client auswählen; für Injection)



Steps	Name and Location	
. Choose Project	Project Name: TaskProducer	
. Server and Settings	Project Location: umente und Einstellungen\Apollo\Eigene Dateien\NetBeansProje	cts Browse
	Project Eolder: Einstellungen\Apollo\Eigene Dateien\NetBeansProjects\TaskPro	ducer
	Use Dedicated Folder for Storing Libraries	
	Libraries Folder:	Browse
	Different users and projects can share the same compilation libraries (see Help for details).	
	🔽 Set as Main Project	
Con a		

New Enterprise Application Cli	ent	×
Steps	Server and Settings	
<ol> <li>Choose Project</li> <li>Name and Location</li> </ol>	Add to Enterprise Application: <a>None&gt;</a>	7
3. Server and Settings	Server: GlassFish Server 3	▼ A <u>d</u> d
	Use dedicated library folder for server JAR files	
	Java EE Version: Java EE 6 💌	
	Enable Contexts and Dependency Injection	
	Main Class: taskproducer.Main	

• Next -> Next -> Finish

o Folgende Projektdateien und -verzeichnisse wurden bisher erzeugt:



#### • JMS-Nachricht versenden

- o Datei , Main. java' öffnen und Leerzeile in main() einfügen.
- Rechts-Click in Leerzeile von main() -> Insert Code



🇊 Send JMS Message		×
O Project Destinations:	×	<u>A</u> dd
Server Destinations:	jms/TaskBagTopic	
C Message Driven Bean:	ClientMDB1	
Destination:	jms/TaskBagOutTopic	
Connection Factory:	jms/TaskBagTopicFactory	
Service Locator Strategy		
	up Code	
C Existing Class		20
	OK Cancel	Help

o OK

• Folgender Code wurde in der Klasse Main erzeugt:

```
public class Main {
      @Resource(name = "jms/TaskBagTopic")
private static Topic taskBagTopic;
      @Resource(name = "jms/TaskBagTopicFactory")
private static ConnectionFactory taskBagTopicFactory;
/**
* @param args the command line arguments
*/
public static void main(String[] args) {
      // TODO code application logic here
}
private Message createJMSMessageForjmsTaskBagTopic(Session session,
      Object messageData) throws JMSException {
      // TODO create and populate message to send
      TextMessage tm = session.createTextMessage();
      tm.setText(messageData.toString());
      return tm;
}
private void sendJMSMessageToTaskBagTopic(Object messageData) throws
            JMSException {
      Connection connection = null;
      Session session = null;
      try {
            connection = taskBagTopicFactory.createConnection();
            session = connection.createSession(false,
                        Session.AUTO_ACKNOWLEDGE);
            MessageProducer messageProducer =
                        session.createProducer(taskBagTopic);
            messageProducer.send(createJMSMessageForjmsTaskBagTopic
                        (session, messageData));
      } finally {
            if (session != null) {
                  try {
                        session.close();
                  } catch (JMSException e)
                        Logger.getLogger(this.getClass().getName()).
                        log(Level.WARNING, "Cannot close session",
                                    e);
            }} if (connection != null) {connection.close()}}}
```

- o Erzeugten Code anpassen:
  - @Resource-Parameter name ersetzen durch mappedName (zwei mal).
  - Die Methoden sendJMSMessageToTaskBagTopic() und createJMSMessage-ForjmsTaskBagTopic() als static deklarieren (wegen static main()).
  - Den Aufruf des Loggers (nicht-static) auskommentieren (wegen static main()).
- o Methode **sendJMSMessageToTaskBagTopic()** in main() aufrufen:
  - Aktuellen Parameter messageData aus dem generierten Code bestimmen und eingeben.
  - JMSException abfangen.

#### • Starten des Task-Producers

• Rechts-Click auf Ordner ,TaskProducer' -> Run.

## TEIL B: Tasks über einen Message Selector an mehrere Task-Consumer senden

### **B.1** Message Selector

- Task-Producer ergänzen
  - Nach der Anweisung setText() ist die Anweisung setStringProperty("taskType", "garten") für die TextMessage auszuführen (mit frei gewähltem Selektortyp "taskType" und frei gewähltem Selektorwert "garten").
  - Nach einem ersten Test mit nur einer Nachricht, ist der Task-Producer weiter zu ergänzen durch eine Nachrichten-Schleife. In jedem Schleifendurchlauf sollen Tasks mit vier verschiedenen Selektorwerten erzeugt und versendet werden. Der Nachrichtentext soll zusätzlich um eine fortlaufende Nummer ergänzt werden, die auf Producer- und Consumer-Seite wieder ausgedruckt wird.
- Task-Consumer ergänzen

  - Die Ausgabe der Textnachricht (mittels getText()) ist zu ergänzen um die Ausgabe des Selektorwertes mithilfe der Anweisung getStringProperty("taskType").
- Weitere Task-Consumer erstellen (mindestens einen weiteren Consumer)
  - o Projektdateien und -verzeichnisse erzeugen (wie A.1).
  - Message Driven Bean erzeugen (wie A.1, außer ,Server Destination' auswählen):

🗊 New Message-Driven Bean	×
Steps	Name and Location
<ol> <li>Choose File Type</li> <li>Name and Location</li> </ol>	E3B Name: TaskConsumer2MDB
	Project: TaskConsumer2
	Location: Source Packages
	Package: taskconsumer2mdb
	C Project Destinations: Add
	< Back Next> Einish Cancel Help

- o Finish.
- Methode *onMessage()* der Message Driven Bean ausprogrammieren (wie A.1).
- Neuen MessageSelector definieren (wie A.1). Dazu die Operatoren *OR*, *IN* und *LIKE* benutzen (siehe Tabelle der Aufgabenstellung, Seite 6).
- Neuen Task-Consumer bereitstellen (wie A.1).
- Tests
  - o Task-Producer mindestens 16 Nachrichten von 4 verschiedenen Task-Typen senden lassen.
  - Message-Selektoren aller Task-Consumer in verschiedenen Varianten durchspielen.
  - Auf der Admin-Konsole unter ,Configuration/Java Message Service/Physical Destinations/Statistics/ View' den Inhalt der Topic prüfen (siehe oben, Seite 4).

## **B.2** Abgekoppelte Task-Consumer

- Abkoppeln aller Task-Consumer <u>mit</u> ActivationConfigProperty "Durable"
  - Auf der Admin-Konsole unter ,Applications' die Task-Consumer (sicherheitshalber einzeln) auswählen und undeployen.

Tree E Common Tasks	Ap	pplications plications can be enterprise	or web applications,	or various kinds of mo	dules.			
- 🚺 Registration - 🚑 GlassFish News - 📋 Enterprise Server	De IV	Deployed Applications (2) (B) Deploy Undeploy Enable Disable   Filter:						
Applications		Name	*	Enabled	1.	Engines	14	Action
- 📄 Lifecycle Modules		TaskConsumer		V		[ejb]		Redeploy   Restart
🔻 🍟 Resources		TaskProducer		<ul> <li></li> </ul>		[appclient]		Redeploy   Restart
<ul> <li>▶ ■ JDBC</li> <li>▶ A Connectors</li> </ul>								

- Starten des Task-Producers (wie A.2). Ausgabe von TaskProducer (run) beachten. Inhalt der Topic prüfen (siehe oben, Seite 4).
- Deploy für Task-Consumer-1 (wie A.1). Ausgabe von GlassFish Server 3 beachten. Inhalt der Topic prüfen (siehe oben, Seite 4).
- o Deploy für Task-Consumer-2, usw. (wie oben).
- Abkoppeln aller Task-Consumer ohne ActivationConfigProperty "Durable"
  - o In allen Task Consumers die Codezeile @ActivationConfigProperty(propertyName =
     "subscriptionDurability", propertyValue = "Durable") auskommentieren.
  - o Danach Test wie oben durchführen und unterschiedliche Ausgaben beachten.