CPSC 476 Problem set 4
Daniel Jordan
daniel_jordan@csu.fullerton.edu

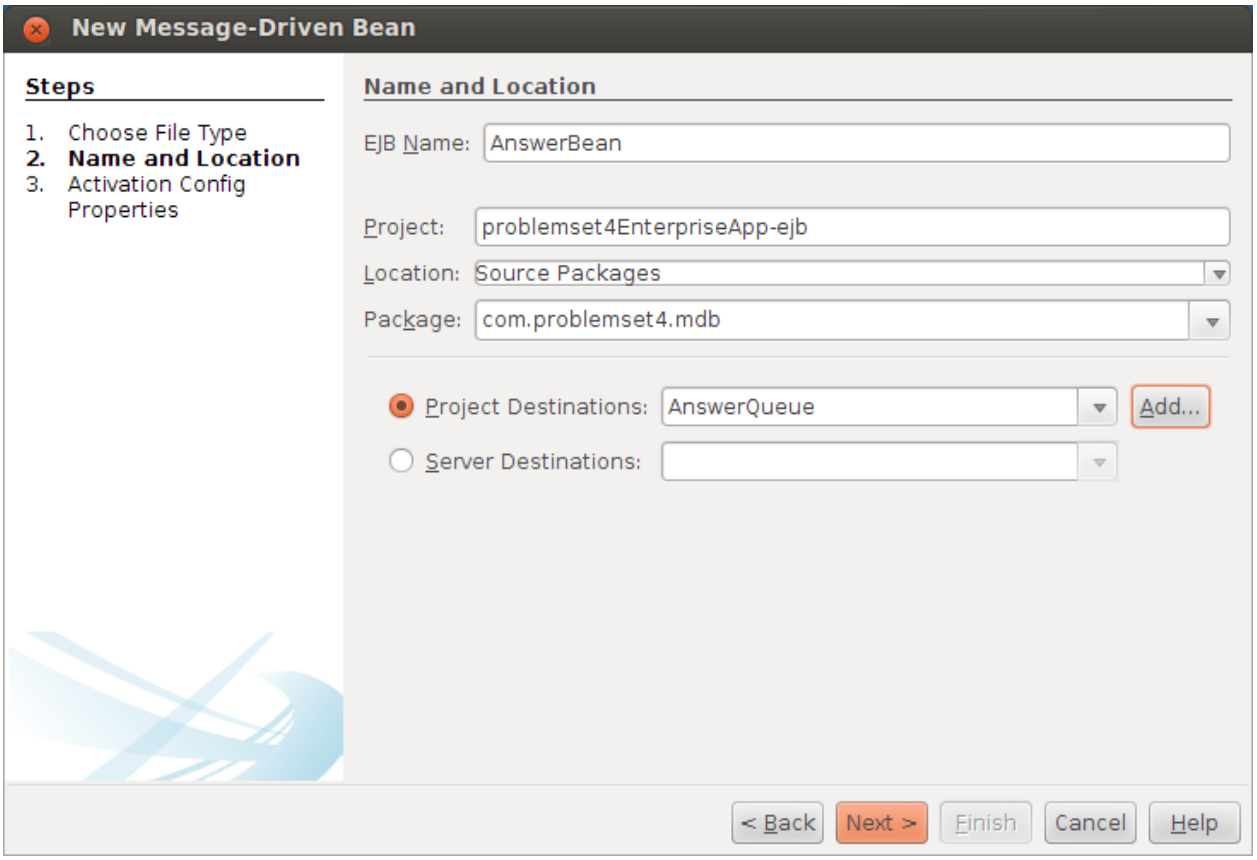
1. Imagine that you plan to write a client to read in text strings in the format given above and create corresponding questions and answers in the database. What issues would you face when trying to use the Answers entity and AnswersFacadeRemote interface created by NetBeans in Problem Set 3?

If I tried to use AnswersFacadeRemote from problem set 3, I wouldn't be able to insert questions, because AnswersFacadeRemote can only read from the database.

2. Create a new Message-Driven Bean called AnswerBean. If you are not adding the MDB to an existing project, be sure to create a new Enterprise Application to contain the EJB module.

To create a message-driven bean, I first created an enterprise application under File > New Project > Java EE > Enterprise Application. I titled the project problemset4EnterpriseApp, and created an ejb module problemset4EnterpriseApp-ejb, to hold the MDB.

To create the MDB, I right clicked on the ejb module, and selected New > Enterprise JavaBeans > Message-Driven Bean. I named the bean AnswerBean, inside the package com.problemset4.mdb, and gave it a project destination "AnswerQueue" of type Queue.



3. In AnswerBean's onMessage() method, retrieve the text from the incoming message, parse it using the format shown above, and create new objects in the database as necessary. Ignore answers for question IDs that do not exist.

To receive the incoming message, I first created a glasshfish resource file, containing the jndi lookup name for the AnswerQueue message queue. Once I am able to connect to the queue, I receive the incoming Message object, verify that it's a TextMessage, and read the string value of the message.

To parse the string, I use the split function, with ":" and "--" used as delimiters in a regex statement. This splits the message into an array with 3 parts: the message type (question, or answer with ID), the message, and the username.

After splitting the message, the MDB connects to the database using JDBC, and searches for the username passed in with the message. To avoid a future foreign key constraint error, non-registered users are inserted into the database, and given a blank password. Another query retrieves the user ID for the new (or existing) user, and stores it in a resultset.

The message type is then looked at. Questions are automatically inserted into the database. For answers, the bean look at the 2nd character of the message type, and converts it to an integer, for the question ID. The answer is then inserted, and the connection to the queue is closed.

4. Create an application client to submit TextMessages to the AnswerQueue.

I created an application client by clicking
File > New Project > Java EE > Enterprise Application Client.

I named the project problemset4EnterpriseClient, and added it to my existing problemset4EnterpriseApp project. From my client, I perform a resource lookup, to find the JMSConnectionFactory and the message queue.

```
public class Main {
    @Resource(lookup = "java:comp/DefaultJMSConnectionFactory")
    private static ConnectionFactory connectionFactory;
    @Resource(lookup = "jms/AnswerQueue")
    private static Destination destination;
```

For each message being sent, a connection is created, a session is created from the connection, a message producer is created from the session, and a TextMessage object is created by the session. The message producer then sends the text message.

```
public static void sendMessage(String message){
    try {
        Connection connection = connectionFactory.createConnection();
        Session session = connection.createSession(false,Session.AUTO_ACKNOWLEDGE);
        MessageProducer messageProducer = session.createProducer(destination);
        TextMessage textMessage = session.createTextMessage(message);
        messageProducer.send(textMessage);
        connection.close();
    }catch (JMSException ex) {
        System.out.println("Error connecting to message queue");
    }
}
```

5. Create an Interceptor DedupeInterceptor for AnswerBean to avoid adding answers to the database that have already been given.

To create the interceptor, I created a new java class, and titled it DedupeInterceptor. I told glassfish this was an interceptor class by adding the @Interceptor annotation just before creating the class. I created the method “interceptMsg” of type Object, and annotated that with @AroundInvoke, to specify AroundInvoke interception.

An InvocationContext object was passed to the method, giving the method control over invoking the message bean. I intercepted the message by calling getParameters() on the InvocationContext object. This returned an array of Message objects. I parsed the message using the same method mentioned above, using the split function with 2 delimiters. For all messages that begin with the letter “A”, I parse the question ID from the message, and check if an answer exists with the same ID and answer body as the incoming message. If the answer exists, the interceptor returns, and the message bean is never invoked. If the answer doesn't exist, the answer is inserted into the database, and the proceed() is called, to move onto the next entry in the interceptor chain.

6. Demonstrate that your application client can add a new answer to the database, and that submitting duplicate answers has no effect.

I hard coded my question/answer transmissions, calling for a 2 second pause between transmissions, to avoid any possible race conditions between messages.

The output below demonstrates the sending of 4 different messages, including one duplicate answer.

Java DB Database Process × | GlassFish Server 4.0 × | problemset4EnterpriseClient (run) × | SQL Command 1 execution ×

```
INFO:   ACDEPL104: Java Web Start services stopped for the app client problemset4EnterpriseClient
INFO:   visiting unvisited references
INFO:   visiting unvisited references
INFO:   ACDEPL103: Java Web Start services started for the app client problemset4EnterpriseClient (contextRoot: /problemset4EnterpriseClient)
INFO:   problemset4EnterpriseClient was successfully deployed in 158 milliseconds.
INFO:   Intecepting message: Q: Why should I study Spring framework if Java ee 7 exists and covers all capabilities which Spring implements? -- puffstone
INFO:   Proceeding to MDB...
INFO:   Received message: Q: Why should I study Spring framework if Java ee 7 exists and covers all capabilities which Spring implements? -- puffstone
INFO:   Question added to database
INFO:   Intecepting message: A2: I think that one simply should know both of them. -- ProfAvery
INFO:   Proceeding to MDB...
INFO:   Received message: A2: I think that one simply should know both of them. -- ProfAvery
INFO:   User ProfAvery added to database
INFO:   Answer added to database
INFO:   Intecepting message: A2: I think that one simply should know both of them. -- dogpockets
INFO:   Answer already exists in database
INFO:   Intecepting message: A2: Lets try another answer then. -- dogpockets
INFO:   Proceeding to MDB...
INFO:   Received message: A2: Lets try another answer then. -- dogpockets
INFO:   User dogpockets added to database
INFO:   Answer added to database
```

I verified the entries, by querying each table of the database after running the program

Users

#	ID	USERNAME	PASSWORD
1	1	fishygut	password
2	2	puffstone	qwerty
3	3	barkingcustard	welcome
4	9	ProfAvery	
5	10	dogpockets	

Questions

#	ID	USER_ID	TEXT	CREATED_AT
1	1	1	What are the main differences between Java EE 7 and Java EE 6 ?	2013-12-09 02:20:17.104
2	2	3	Where can I find code examples for Java 7 EE Tutorial?	2013-12-09 02:20:17.128
3	12	2	Why should I study Spring framework if Java ee 7 exists and covers all capabilities which ...	2013-12-09 17:09:59.943

Answers

#	ID	USER_ID	QUESTION_ID	TEXT	CREATED_AT
1	1	2	1	Support for JSON	2013-12-09 02:20:17.165
2	2	3	1	GlassFish v4	2013-12-09 02:20:17.202
3	3	2	1	Improved Bean Validation	2013-12-09 02:20:17.229
4	4	3	1	WebSocket support	2013-12-09 02:20:17.252
5	5	2	2	https://java.net/projects/javaeetutorial/sources/svn/s...	2013-12-09 02:20:17.294
6	6	3	2	Thanks!	2013-12-09 02:20:17.313
7	15	9	2	I think that one simply should know both of them.	2013-12-09 17:10:02.06
8	16	10	2	Lets try another answer then.	2013-12-09 17:10:06.092