//------------Create an EJB application

Create java EE => Enterprice application

Right click on EJB module => new => other => persistence => Entity class

private String title;

private String body;

//----------------generate getters and setters

Right click on EJB module => new => other => Eneterprice javaBeans => MessageDrivenBean

Select the package name ejb => select the add button next to the project destination

jms/NewMessage => to the Name select ok => finish

public class NewMessageBean implements MessageListener {

@Resource

private MessageDrivenContext mdc;

Then add those two lines

Right click the code and select insert code ? Use Entity Manager

Change the persist => save

//-------------------Then add following codes to the onMessage method

public void onMessage(Message message) {

ObjectMessage msg = null;

try {

if (message instanceof ObjectMessage) {

msg = (ObjectMessage) message;

NewsEntity e = (NewsEntity) msg.getObject();

save(e);

}

} catch (JMSException e) {

e.printStackTrace();

mdc.setRollbackOnly();

} catch (Throwable te) {

te.printStackTrace();

}

}

//-------------------------------------------------------------------

Right click the code then

EJB module => new => other => persistence => sessionBean for entity class

addAll => finish

it will create 2 classes AbstractFacade class and newsEntityFacade

//---------right click the web project => new => other => java Beans => select Sinalton => finish (package ejb)

//-------------------------add

@Singleton

@LocalBean

@WebListener

//--------------------------------

Here

public class SessionManagerBean implements HttpSessionListener {

implement the class HttpSessionListener

then resolve the error using implement all abstract methods

public class SessionManagerBean implements HttpSessionListener{

private static int counter = 0;

include the private static int counter = 0; here

counter increment and decrement here

public void sessionCreated(HttpSessionEvent se) {

counter++;

public void sessionDestroyed(HttpSessionEvent se) {

counter--;

public int getActiveSessionsCount() {

return counter;

}

Add the method to get the counter

Creating the ListNews Servlet

Create the servlet => new => servlet => package =>web => finish

Right click the servlet => insert code => Call Enterprise Bean.

In the Call Enterprise Bean dialog box, expand the NewsApp-ejb node and select

NewsEntityFacade. Click OK.

Use the Call Enterprise Bean dialog box again to inject the SessionManagerBean under the

NewsApp-war node.

In the processRequest method, add the following code to return the current

session or create a new one. request.getSession(true);

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.getSession(true);

Add the following code to the processRequest method to print the messages and

add a link to the PostMessage servlet. (Uncomment the code in the method if necessary.)

"</h1>");

List news = newsEntityFacade.findAll();

for (Iterator it = news.iterator(); it.hasNext();) {

NewsEntity elem = (NewsEntity) it.next();

out.println(" <b>"+elem.getTitle()+" </b><br />");

out.println(elem.getBody()+"<br /> ");

}

out.println("<a href='PostMessage'>Add new message</a>");

out.println("</body>");

Add the following code (in bold) to retrieve and print the number of users/open sessions

out.println("<a href='PostMessage'>Add new message</a>");

out.println("<br><br>");

out.println(sessionManagerBean.getActiveSessionsCount() + " user(s)reading the news.");

out.println("</body>");

Creating the PostMessage Servlet

Right-click the web module project and choose New > Servlet

Enter web for the Package name and click Finish.

Use annotations to inject the ConnectionFactory and Queue resources by adding the

following field declarations

@WebServlet(name = "PostMessage", urlPatterns = {"/PostMessage"})

public class PostMessage extends HttpServlet {

@Resource(mappedName = "jms/NewMessageFactory")

private ConnectionFactory connectionFactory;

@Resource(mappedName = "jms/NewMessage")

private Queue queue;

You now add the code to send the JMS messages by adding the following code in bold to

the processRequest method:

response.setContentType("text/html;charset=UTF-8");

// Add the following code to send the JMS message

String title = request.getParameter("title");

String body = request.getParameter("body");

if ((title != null) && (body != null)) {

try {

Connection connection = connectionFactory.createConnection();

Session session = connection.createSession(false,

Session.AUTO\_ACKNOWLEDGE);

MessageProducer messageProducer

= session.createProducer(queue);

ObjectMessage message = session.createObjectMessage();

// here we create NewsEntity, that will be sent in JMS message

NewsEntity e = new NewsEntity();

e.setTitle(title);

e.setBody(body);

message.setObject(e);

messageProducer.send(message);

messageProducer.close();

connection.close();

response.sendRedirect("ListNews");

} catch (JMSException ex) {

ex.printStackTrace();

}

}

try (PrintWriter out = response.getWriter()) {

if you get any error messages that you have to check the import statement that has bean used

it should be

import javax.jms.Connection; for the sql connection

import javax.jms.Queue; for que

Add the following lines (in bold) to the processRequest method to add the web form for

adding a message. (Uncomment the code to print the HTML if necessary.)

out.println("<h1>Servlet PostMessage at " + request.getContextPath() + "</h1>");

// The following code adds the form to the web page

out.println("<form>");

out.println("Title: <input type='text' name='title'><br/>");

out.println("Message: <textarea name='body'></textarea><br/>");

out.println("<input type='submit'><br/>");

out.println("</form>");

out.println("</body>");

Running the Project

In the Projects window, right-click the NewsApp enterprise application node and select

Properties in the pop-up menu.

Select Run in the Categories pane.

In the Relative URL textfield, type /ListNews.

Click OK.